UNIVERSIDAD DE EL SALVADOR

FACULTAD DE INGENIERIA Y ARQUITECTURA

ESCUELA DE INGENIERIA DE SISTEMAS INFORMATICOS

SISTEMA INFORMÁTICO PARA LA GESTIÓN Y CONTROL DE LAS ACTIVIDADES DE LA FEDERACIÓN SALVADOREÑA DE SURF (SWELL)

PRESENTADO POR:

ALEXANDER WILFREDO GUEVARA ESCALANTE
LORENZO VICENTE GRIMALDI VELASCO
MILTON ALEXANDER SOSA FIGUEROA

PARA OPTAR AL TITULO DE:

INGENIERO DE SISTEMAS INFORMATICOS

CIUDAD UNIVERSITARIA, NOVIEMBRE DE 2018

UNIVERSIDAD DE EL SALVADOR

RECTOR:

MSc. ROGER ARMANDO ARIAS ALVARADO

SECRETARIO GENERAL:

MSc. CRISTOBAL HERNAN RIOS BENITEZ

FACULTAD DE INGENIERIA Y ARQUITECTURA

DECANO:

ING. FRANCISCO ANTONIO ALARCON SANDOVAL

SECRETARIO:

ING. JULIO ALBERTO PORTILLO

ESCUELA DE INGENIERIA DE SISTEMAS INFORMATICOS

DIRECTOR:

ING. JOSE MARIA SANCHEZ CORNEJO

UNIVERSIDAD DE EL SALVADOR

FACULTAD DE INGENIERIA Y ARQUITECTURA

ESCUELA DE INGENIERIA DE SISTEMAS INFORMATICOS

Trabajo de Graduación previo a la opción al Grado de:

INGENIERO DE SISTEMAS INFORMATICOS

Título:

SISTEMA INFORMÁTICO PARA LA GESTIÓN Y CONTROL DE LAS ACTIVIDADES DE LA FEDERACIÓN SALVADOREÑA DE SURF (SWELL)

Presentado por:

ALEXANDER WILFREDO GUEVARA ESCALANTE

LORENZO VICENTE GRIMALDI VELASCO

MILTON ALEXANDER SOSA FIGUEROA

Trabajo de Graduación Aprobado por:

Docente Asesor:

ING. JOSE MARIA SANCHEZ CORNEJO

SAN SALVADOR, NOVIEMBRE DE 2018

Trabajo de Graduación Aprobado por:

Docente Asesor:

ING. JOSE MARIA SANCHEZ CORNEJO

Contenido

I. Introducción

En el presente documento, se aborda el desarrollo de un SISTEMA INFORMÁTICO PARA LA GESTIÓN Y CONTROL DE LAS ACTIVIDADES DE LA FEDERACIÓN SALVADOREÑA DE SURF (SWELL), el cual busca mejorar la eficiencia en el procesamiento de la información en las actividades de la federación, con el fin que el personal involucrado pueda mejorar el desempeño en sus labores, Impulsar el éxito del deporte del Surf en nuestro país y reconocer el talento nacional seleccionando quien será o quienes nos representarán en competencias a nivel internacional.

Se definen y plantean los alcances, así como la importancia del desarrollo e implementación del sistema informático como solución, para mejorar y facilitar los procesos que se ejecutan diariamente en dicho ente deportivo, también se ha elaborado una breve justificación con énfasis en la gestión de grandes volúmenes de información de forma manual, que tienden a disminuir la productividad del personal. Por otra parte, se adoptó la metodología RUP para el desarrollo del sistema informático debido a, que está diseñada para asegurar la producción de software de alta calidad para satisfacer las necesidades de los usuarios.

Asimismo, se expone un estudio preliminar que comprende de los antecedentes y estructura organizativa de la Federación Salvadoreña de Surf; Además, mediante el diagrama de estado o caja negra se define la problemática central a solventar, y con el diagrama de causa y efecto, se identifican las causas principales que dan como resultado la deficiencia en la gestión administrativa de información. El uso de herramientas como la entrevista formó parte fundamental para realizar el análisis de los requerimientos del sistema, con el fin de satisfacer las necesidades de los usuarios; también, se definieron estándares de trabajo en el ámbito de diseño, programación y documentación, con el propósito de estructurar la construcción del sistema.

En el diseño del sistema se han tomado en cuenta aspectos muy importantes como el modelo arquitectónico, el diseño de interfaces de usuario y seguridad del sistema. Una vez contemplado cada detalle que involucrará el sistema informático se ha realizado el diseño de la base de datos, esta dará soporte al almacenamiento de la información capturada y generada por los usuarios, estratégicamente se ha diseñado una base de datos por esquemas que facilitan su comprensión y respectiva documentación en el diccionario de datos. La documentación de un sistema informático es muy importante, pues facilitara el uso adecuado y modificación de las funciones de este. Finalmente se detalla un plan de pruebas a seguir para asegurar un sistema libre de errores y a su vez aceptado por el usuario, además de un plan de implementación que garantice la puesta en funcionamiento del sistema informático.

II. Objetivos

Objetivos General

Optimizar y Mejorar el procesamiento de los datos en los procedimientos de gestión y control de las actividades de la Federación Salvadoreña de Surf así como también de los recursos que se poseen.

Objetivos Específicos

- Analizar la situación actual de la administración para identificar procedimientos e información relevante que intervenga y sea de gran valor para el desarrollo del sistema informático.
- Analizar los requerimientos del Sistema Informático mediante el levantamiento, validación, verificación y documentación de los mismos.
- Diseñar una solución de Sistema Informático que proporcione un mejor control de los procedimientos actuales.
- Construir un Sistema Informático basado en el diseño previo de la solución.
- Elaborar y ejecutar un plan de pruebas para garantizar un Sistema Informático funcional, libre de errores y aceptado por el usuario.
- Elaborar la documentación del Sistema Informático, incluyendo los manuales de instalación, manual de Usuario y Manual Técnico.
- Elaborar un plan de implementación que garantice la puesta en función del Sistema informático para la gestión y control de las actividades de la federación salvadoreña de surf (SWELL).

III. Alcances

El Sistema Informático que apoyará el registro de datos relacionados incluirá los siguientes módulos:

- Módulo que permita registrar atletas, entrenadores, jueces, clubes, escuelas, y los miembros de junta directiva además de los patrocinadores
- Módulo que permita el manejo de las competencias, calendarización y que permita al público consultar los estados y resultados de las mismas, además de poder llevar un histórico sobre las competencias y puntuaciones de los surfistas.
- Módulo que permita llevar registro y control, de entradas y salidas de efectivo, estas incluyen las donaciones hechas por patrocinadores.
- Módulo que permita dar a conocer las actividades próximas a realizarse y que permita incluir un afiche de referencia.
- Aplicación móvil que permita tanto a sufistas como aficionados recibir notificaciones de competencias y poder visualizar resultados de las mismas en tiempo real, así como notificaciones de actividades a realizarse.
- Además se incluirá un Sitio Web Publico en el que todas las personas interesadas podrán consultar la información autorizada por la federación.

Al finalizar este proyecto se entregará un Sistema Informático libre de errores, aprobado por el usuario y listo para ser implementado, cumpliendo los requerimientos del usuario generando procesos eficientes.

El Sistema Informático será entregado con la siguiente documentación:

- Especificaciones del análisis y diseño del sistema informático.
- Manual de Instalación.
- Manual de Usuario.
- Manual Técnico.
- Plan de Implementación.
- Plan de Pruebas

Para efectos de probar el correcto funcionamiento del Sistema Informático, este será instalado y ejecutado en la nube.

IV. Limitaciones

Haciendo una síntesis de cada uno de los resultados antes mencionados, para la realización del presente proyecto, se concluye que no existe ninguna limitación que perjudique su realización

V. Importancia

- Reducción de los tiempos para toma de decisiones en las actividades de la FESASURF ya que serán disminuidos considerablemente, al contar con resúmenes de los datos guardados e historiales.
- Eficiencia en la gestión de los recursos que son asignados a la FESASURF, ingresos y gastos, los cuales son asignados por el INDES, y las aportaciones hechas por patrocinadores.
- Eficiencia en el procesamiento de datos en las competiciones, de una manera rápida que permita saber los resultados en tiempos cortos.
- Dar a conocer el talento nacional en el surf y el reconocimiento de este deporte a nivel nacional e internacional con apoyo de un sistema de información que facilite el día a día en las actividades de la FESASURF, con el fin de generar mecanismos de selección sobre quien será o quienes nos representarán en competencias a nivel internacional.
- Aumento del turismo en la zona costera ya que se cuenta con playas ideales para practicar este deporte extremo, lo cual atrae muchos surfistas de diversas partes del mundo, convirtiendo al país en un destino turístico.

VI. Justificación

- En el 2016 la FESASURF cerró el año con 135 atletas federados. Consultar la información de los mismos es un proceso muy lento ya que estos datos se almacenan en fichas.
- Como Federación es importante que esta lleve registros de cada una de las 6 competencias que se realizan cada año. En cada fecha del circuito nacional de surf se inscriben más de 150 participantes, los datos que se generan servirán tanto para el ranking nacional, así como para compartir la información con el INDES.
- Se tendrá un mayor control de cuántas escuelas tiene la FESASURF, actualmente se cuenta con una escuela oficial, pero existen más de 20 escuelas no oficiales de surf.
- Mejorar el resguardo de la información ya que contará con un respaldo de la información por lo que será fácil acceder a datos históricos.
- En un mundo actualizado y siendo el surf un deporte que llama la atención, fácilmente puede incrementar el turismo en la zona costera, por ende la importancia de manejar un calendario con cada una de las actividades.
- Los surfistas tendrán mayor información de las competencias por tanto se espera una mayor participación y crecimiento en el deporte. Y aumentando la probabilidad de ganar medallas.
- El surf es un deporte en el que es importante tener una plataforma que pueda dar información sobre las actividades que se realizan.
- Se mostrarán resultados en tiempo real de las competiciones mediante la implementación de una aplicación para dispositivos móviles, y con esto aumentar el alcance de FESASURF.
- Se implementará mediante el uso de TICS, la modalidad de Transparencia Activa donde se llevará un control total de los ingresos y egresos y cualquier persona podrá tener acceso a esa información.

1. CAPÍTULO I: ESTUDIO PRELIMINAR

1.1. Antecedentes

1.1.1. Historia

Desde hace más de 12 años fue fundada la Federación Salvadoreña de Surf (FESASURF) como una organización sin fines de lucro que es reconocida por el Instituto Salvadoreño de los Deportes (INDES), y por el Comité Olímpico de El Salvador (COES) como organismo rector del surf en El Salvador respondiendo a este deporte en auge.

La Federación ha tenido problemas en lo que respecta al manejo de la información, ya que los volúmenes de información van desde los datos personales de los surfistas, seguimiento de las escuelas y competencias en la que participan así como de igual manera los clubes y escuelas, una de las dificultades que se ha registrado también es la gestión de la información financiera ya que se desea poder gestionar de manera más adecuada las entradas y salidas de efectivo, permitiendo clasificarlas y poder visualizar dicha información de manera más sencilla.

Históricamente el proceso a seguir si un surfista quiere registrarse como federado, ha sido llegar a la federación y llenar un formulario con sus datos personales, además de información pertinente sobre el tiempo que tiene de practicar el deporte, haciendo valido de esta manera su registro, de igual forma debe de registrarse nuevamente para poder participar en alguna competición, la gestión de la información anterior se realiza de forma manual en papel o utilizando hojas de cálculo para consolidar los perfiles de los surfistas.

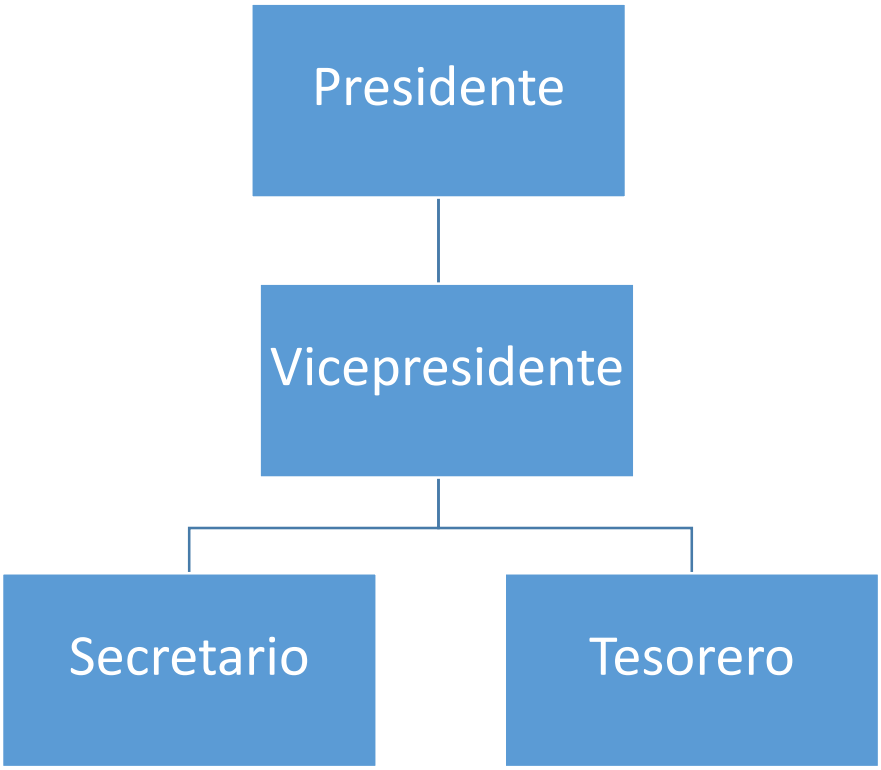
Dentro de la información más relevante que se maneja incluye saber sobre cada una de los clubes que están registrados bajo la FESASURF, saber dónde están ubicados y quien es el representante.

También se maneja el registro detalle de lo que ingresó contra el registro detalle de lo que egresó, todo por facturación y al final se hace un consolidado presentando una rendición de cuentas anual al INDES para cuadrar, abonando fondos sobrantes a la federación.

Adicionalmente se debe gestionar los méritos que tiene cada atleta federado, algo de suma importancia para la FESASURF ya que es necesario reconocer el talento nacional e impulsar al éxito de este deporte en nuestro país, con el fin de seleccionar quien será o quienes nos representarán en competencias a nivel internacional.

Es importante también para la FESASURF los campeonatos que se realizan tanto con surfistas locales como con surfistas internacionales (Actualmente son 6 fechas durante el año), los cálculos de los puntajes de cada competencia se realizan de manera manual (se suman los puntos y quienes obtengan el top 3 pasan a la siguiente ronda). Esto genera una demora prolongada en la entrega de resultados, ya después de tener los resultados es necesario pasar esta información a archivos de Excel para poder tener un consolidado al final de cada competencia.

1.1.2. Estructura Organizativa FESASURF



1.2. Definición de Problema

1.2.1. Formulación del Problema

Situación actual

FESASURF año con año va aumentando el volumen de información y hasta el momento no había existido la necesidad de implementar un sistema que llevara el control de dicha información, pero al ser El Surf un deporte en auge aumentan los datos de atletas, entrenadores, escuelas de surf, clubes deportivos y competencias que se manejan.

Lo Anterior ocasiona un problema ya que FESASURF registra toda la información de manera manual, mediante el uso de archivos físicos y mediante el uso de Hojas de cálculo, esto debido a que la información que se manejaba era poca con respecto a la que se tiene actualmente, lo cual dificulta el procesamiento de dicha información.

FESASURF realiza varias tareas utilizando este procedimiento como las mencionadas a continuación:

Gestión de Atletas:
Cada surfista debe de estar registrado en la FESASURF para poder participan en los circuitos de surf que se realizan durante el año, mediante estos registros la federación gestiona los perfiles de cada surfista federado y maneja las puntuaciones por torneo y un acumulado al final de año. Esta información sirve también para el ranking de atletas, y mediante esto se puede seleccionar a las personas que representarán al país en competencias internacionales.

Gestión de Entrenadores:
Así también se lleva registro de las personas que pueden fungir como entrenadores de surf

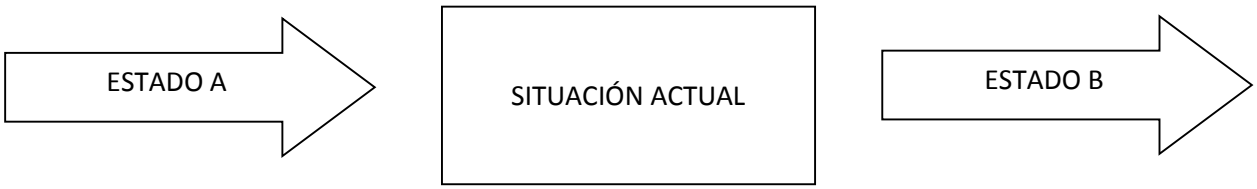
Gestión de Clubes:
FESASURF lleva registro sobre los clubes de surf que existen a lo largo de la línea costera salvadoreña, esto es importante porque necesitan gestionar el presupuesto que se les asigna año con año y que depende del gobierno central.

Gestión de Campeonatos:
Todos los años FESASURF realiza un aproximado de 6 circuitos de surf, en los cuales participan 135 atletas individuales, tanto surfistas nacionales como internacionales, de los cuales es necesario llevar registro sobre las puntuaciones por cada nivel de competencia y de cada uno de los surfistas participantes.

Gestión de Jueces
Es necesario llevar registro de quienes fueron los jueces que asignaron las puntuaciones ya que por cada competencia son 5 jueces los que evalúan.

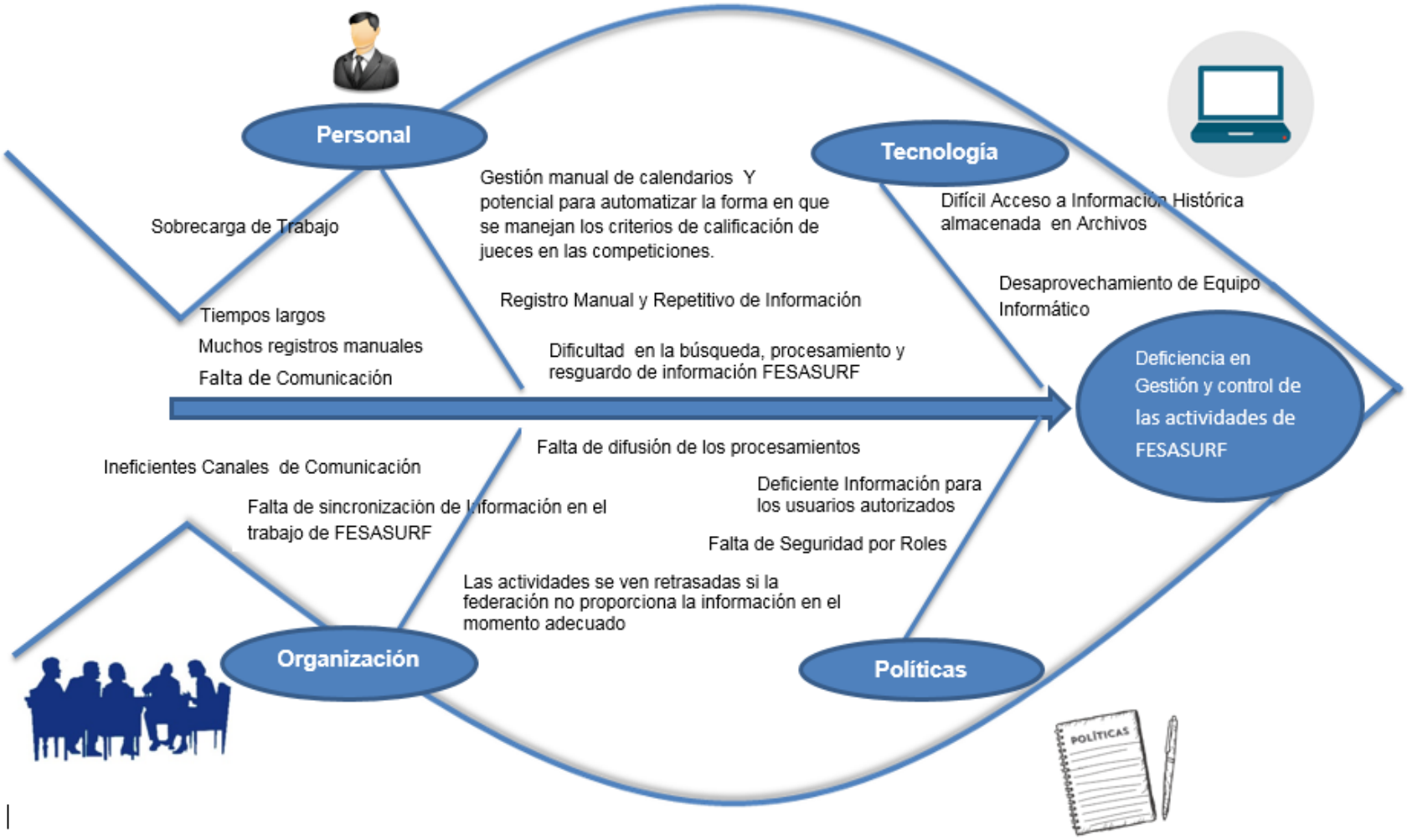
Gestión de fondos:
FESASURF tiene asignado un presupuesto aproximado de $25,000 por año, de los cuales al final de cada año debe de dar un informe de la ejecución de este. También la FESASURF recibe donaciones tanto de patrocinadores como personas naturales y hay que llevar registro detalle de esta información.

1.2.1.1. Diagrama A-B (Estado Actual – Estado Deseado)



• **Dificultad en la búsqueda, procesamiento y resguardo de información FESASURF.** • **Las actividades se ven retrasadas si la federación no proporciona la información en el momento adecuado.** • **Deficiencia en el control y registro de recursos, gastos incurridos.** • **Gestión manual de calendarios Y potencial para automatizar la forma en que se manejan los criterios de calificación de jueces en las competiciones.**	**SISTEMA INFORMÁTICO PARA LA GESTIÓN Y CONTROL DE LAS ACTIVIDADES DE LA FEDERACIÓN SALVADOREÑA DE SURF**

1.2.1.2. Diagrama Causa y Efecto de la Situación Actual



1.3. Planificación de Recursos

A continuación se detalla la planificación de los recursos e insumos en los que se incurrió para realizar el presente proyecto y el presupuesto estimado necesario para el desarrollo del mismo

Recurso Humano

El recurso humano cuenta con las habilidades requeridas para el desarrollo de este proyecto

Recursos	Cantidad	Salario Mensual	Tiempo(Meses)	Total($)
Jefe de Proyecto	**1**	**1200**	**18**	**$21,600**
Programadores	**3**	**900**	**18**	**$48,600**
Total				**$70,200**

Costos totales empleados por el equipo de desarrollo en los 18 meses= **$70,200**

Recursos Tecnológicos

Los recursos de tecnología indispensables para la realización de este proyecto se detallan a continuación.

Cálculo de depreciación NIC 16 (párrafo 6 – Depreciación – Método Línea Recta):

Costo inicial de la laptop=**$500**

Costo final estimado =**$80**

Valor depreciable=**500-80=420**

Vida útil=**5 años**

Depreciación=Valor depreciable/vida útil = **420/5 = $84.00** (dólares/año)

Recursos	Cantidad	Costo($)	Total($)
Laptops	**4**	**84**	**336**
Memorias USB	**4**	**8.10**	**32.40**
Impresoras	**2**	**35.60**	**71.20**
Total			**$439.6**

Costo total en laptops= laptops * Depreciación = **4 * 84 =$336** (dólares/año)

Costo total utilizado en recursos tecnológicos=**$439.6**

Recursos Materiales

Recursos	Cantidad	Costo Unitario($)	Total($)
Fotocopias	**600**	**0.02**	**12.00**
Folder	**19**	**0.15**	**2.85**
Anillado	**10**	**5**	**50.00**
Tinta de color	**2**	**15**	**30.00**
Tinta negra	**2**	**12**	**24.00**
Empastado	**3**	**20**	**60.00**
Resma de papel	**4**	**3.80**	**15.20**
Renta de cañón	**4**	**6**	**24.00**
Total			**$218.05**

Costo total de los materiales= **$218.05**

Transporte

Lugar	Pasaje requerido($)	Frecuencia Semanal	Costo de Pasaje($)	Sub-Total($)
UES (Asesoría)	**8**	**2**	**0.20**	**3.20**
FESASURF	**8**	**1**	**0.35**	**2.80**
Reuniones Grupo	**8**	**5**	**0.20**	**8.00**
Total				**14.00**

Costo mensual en transporte= $14.00 (dólares/semanal) * 4 (semanas) = $56.00

Costo mensual en Transporte por 18 meses= $56.00 (dólares/mensual) * 18 = $1,008.00

Costo total Transporte= **$1,008.00**

Servicios

Los costos de servicios descritos son los que se tendrán a lo largo de los 18 meses de desarrollo.

Recursos	Meses	Costo Mensual($)	Monto($)
Energía Eléctrica	18	40.00	720.00
Internet	18	30.00	540.00
Agua	18	12.00	216.00
Total			1,476.00

Viáticos y Otros

Recursos	Monto($)
Alimentación	904.00
Imprevistos	2,864.39
Total	3,768.39

Resumen

Recursos	Monto ($)
Humano	70,200.00
Tecnológico	439.60
Materiales	218.05
Transporte	1,008.00
Servicios	1,476.00
Viáticos y Otros	3,768.39
TOTAL	77,110.04

Como resumen de la planificación de recursos se muestra el valor del costo para desarrollar el proyecto en 18 meses con un monto de **$77,110.04**

Costo Total de los Recursos del Proyecto en 18 meses = $77,110.04

2. CAPITULO II: ANÁLISIS Y DETERMINACIÓN DE REQUERIMIENTOS

2.1. Enfoque del Sistema Propuesto

Objetivo del sistema.

Automatizar y mejorar el procesamiento de la información generada por en la Federación Salvadoreña de Surf, esto con el fin para brindar un servicio de calidad que aumente la eficiencia y el alcance de la FESASURF, estas mejoras traerán consigo una reducción considerable de los tiempos de respuesta, además de ser capaz de generar información relevante para la toma de decisiones tanto la que está relacionada a surfistas, como la que está relacionada a la parte administrativa.

Descripción de salidas

- Informes de competencias: para las competencias se requiere la creación de los siguientes datos específicos
    - Datos de competencias: contiene las competencias que se han realizado su fecha de inicio y su posible fecha de finalización tanto como el número de surfistas participantes en ella.
    - Datos de surfistas por competencias: permite visualizar todos los surfistas que están inscritos en una competencia, así como sus puntuaciones.
    - Datos por fechas: contiene todas las competencias realizadas en un periodo determinado de tiempo y ver todas sus fechas.
    - Ranking de surfista: permite listar por criterios de participación en las competencias el ranking anual de los surfistas.
- Perfiles de surfistas
    - Datos generales de surfistas: Datos básicos que contiene un número de identificación de surfista (campo sugerido para la identificación de los mismos) así como su nombre, apellido edad y número de competencias en las que ha participado.
    - Gastos en surfistas: Permite ver los gastos que se le hacen a cada surfista se debe generar un reporte general con todos los surfistas y datos resumidos de gastos y un reporte de cada surfista al que se le han dado patrocinios con el registro del detalle de los mismos.
- Informe de Ingresos y Gastos
    - Reporte de entradas y salidas de efectivo: permite visualizar cada una de las entradas de efectivo y gastos de la federación en un periodo determinado de tiempo.
    - Permite generar un reporte de Ingresos y gastos actual de la federación de surf.
- Escuelas y clubes.
    - Informe detallado de las escuelas y los clubes existentes, así como sus integrantes (Atletas, entrenadores).

Descripción de entradas

- Registros de surfistas: Contiene las fichas con datos de los surfistas, así como sus datos relevantes de las competencias en las que ha participado.
- Escuelas y clubes: Contiene las fichas de cada escuela, con la información pertinente de cada una, en la que está incluida la ubicación así como el nombre del representante.
- Puntajes de competiciones: fichas y hojas de cálculo donde se guarda los datos de los participantes de la competición y los resultados de las mismas que son las tablas de puntuaciones

- Ingresos y Gastos: Son fondos que aporta INDES ($25,500 aproximadamente) y donaciones: de los patrocinadores.

Descripción de proceso

- Módulo de perfiles Personales:
  - Registro de jueces por cada competencia durante el año.
  - Captura de datos de los Atletas.
  - Captura de datos de Jueces
  - Captura de datos de Entrenadores.
  - Captura de datos de Miembros de junta.
  - Captura de datos de Patrocinadores (Personas Naturales)
  - Captura de datos de Clubes y Escuelas.
- Módulo de Competencias:
  - Registro de jueces por cada competencia durante el año.
  - Registro de Atletas porcada competencia durante el año.
  - Captura de datos por cada categoría.
  - Captura de puntos de cada juez durante cada una de las rondas, este proceso es para cada uno de los atletas que participan.
  - Calculo de puntos para ranking general, este proceso es para cada uno de los atletas registrados.
- Módulo de Seguridad:
  - Captura de datos para usuarios que tendrán accesos al sistema.
  - Asignación de permisos por roles (niveles de acceso)
- Módulo de Ingresos y Gastos:
  - Captura de Ingresos y gastos de dinero, tanto atletas y patrocinadores. El nivel de registro será mediante transacciones y uso de cuentas de bancos.
- Módulo de noticias
  - Captura de contenido de interés referente al surf, así como contenido referente a la federación.

Control.

- Estatutos y Bases de Competencia de surf:
  - Como toda rama del deporte el surf tiene sus propias reglas para practicar este deporte.
- Normativas de INDES y COES:
  - La FESASURF es la encargada de administrar los recursos humanos y económicos para realizar el deporte, pero esta debe de cumplir con las normas dictadas por el INDES y el COES. El INDES puede programar auditorias para poder verificar que la información presentada es válida y no presenta incongruencias. Por su parte el COES también tiene estatus y ellos son los encargados de gestionar las salidas de atletas a competencias a nivel internacional.

Medio Ambiente.

- INDES: Ente rector del gobierno que controla cada federación a nivel nacional
- COES: Ente rector que controla que federaciones participan en competencias a nivel internacional
- Surfistas: son todas las personas que interactúan directamente con la FESASURF
- Patrocinadores: Son personas o instituciones que aportan económicamente a la FESASURF

Frontera.

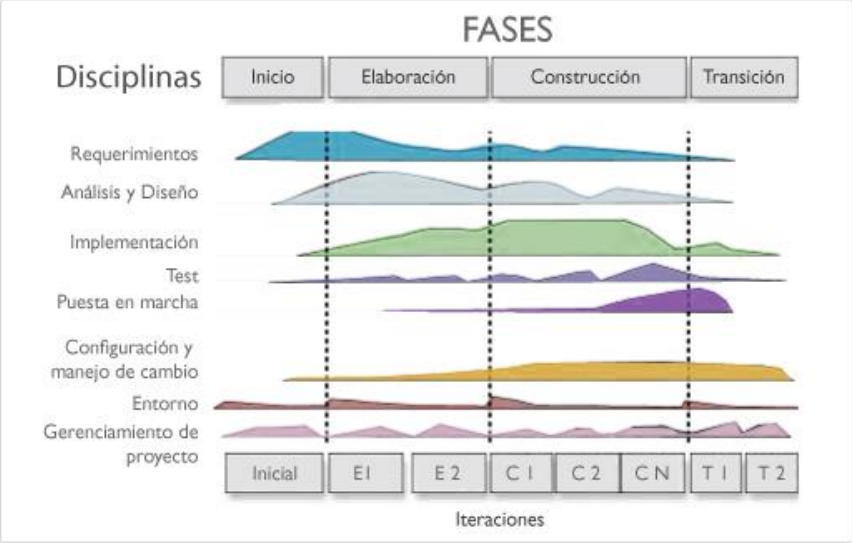
Gestión y control de las actividades de la Federación Salvadoreña de Surf

2.2. Metodología

Después de realizar una investigación de las tres metodologías que más se adaptan al contexto del proyecto (ver anexos) se decidió de elegir RUP. Porque a pesar que ninguna metodología cumple las al cien por ciento las necesidades del proyecto RUP demostró ser la que más se adapta al mismo.

El Rational Unified Process o Proceso Unificado de Racional. Es un proceso de ingeniería de software que suministra un enfoque para asignar tareas y responsabilidades dentro de una organización de desarrollo. Su objetivo es asegurar la producción de software de alta y de mayor calidad para satisfacer las necesidades de los usuarios que tienen un cumplimiento al final dentro de un límite de tiempo y presupuesto previsible. Es una metodología de desarrollo iterativo que es enfocada hacia diagramas de los casos de uso, y manejo de los riesgos y el manejo de la arquitectura como tal.

El RUP mejora la productividad del equipo ya que permite que cada miembro del grupo sin importar su responsabilidad específica pueda acceder a la misma base de datos incluyendo sus conocimientos. Esto hace que todos compartan el mismo lenguaje, la misma visión y el mismo proceso acerca de cómo desarrollar un software. **Las fases de RUP son las siguientes: Inicio, Elaboración, Construcción y Transición.**



Implementación de RUP para el Proyecto

La metodología RUP es más apropiada para proyectos grandes (Aunque también pequeños), dado que requiere un equipo de trabajo capaz de administrar un proceso complejo en varias etapas. En proyectos pequeños, es posible que no se puedan cubrir los costos de dedicación del equipo de profesionales necesarios.

Fase de inicio (Requerimientos)

Para la fase de inicio se deberá tener los siguientes documentos entregables, se estima un mes para esta fase:

- **Solicitudes del cliente.** la importancia de este documento contiene todas las peticiones hechas sobre el proyecto, y el enfoque o posible respuesta a las mismas.
- **Visión del proyecto de software.** el propósito de este documento es recolectar, analizar y definir las necesidades y características del proyecto a un alto nivel. Este se enfoca a las necesidades requeridas por las solicitudes de los clientes.
- **Plan de desarrollo de software.** el plan de desarrollo de software describe los principales elementos del plan de trabajo, como son:
    - Etapas de desarrollo y sus correspondientes fechas de terminación.
    - Recursos involucrados en el desarrollo del proyecto.
    - Productos del sistema con sus respectivas fechas de entrega.
    - Estructura del equipo de trabajo.
    - Plan de trabajo con sus respectivos tiempos.
- **Lista de riesgos** el propósito del documento de lista de riesgos es identificar y evaluar riesgos: con base a la visión del proyecto inicial; identificar, analizar y dar prioridad a los riesgos del proyecto para determinar las estrategias de gestión de riesgos apropiadas.
- **Plan de iteración.** este documento se realiza con el propósito de que el administrador de proyecto pueda planear las actividades y tareas de iteración, detectar los recursos necesarios y tener un registro del progreso. Además, los miembros del equipo de trabajo lo utilizan para saber qué actividades deben realizar, por qué y cuáles actividades dependen de las que ellos están realizando.
    - **Requerimientos técnicos.** Requerimientos del sistema se realiza una lista los requerimientos necesarios para soportar la aplicación, este puede incluir:
    - El sistema operativo.
    - Plataformas para trabajar.
    - Configuración memoria.
    - Programas compartidos.
    - Requerimientos de desempeño.

Se presenta también lista de los requerimientos de desempeño que describen la capacidad de:

- Comunicación
- Rendimiento
- Exactitud
- fiabilidad
- Tiempos de respuesta.

- **Revisiones con el cliente.** Se programan las revisiones con el cliente para revisión de avances del proyecto, se espera que estas revisiones serán cada dos o tres semanas.
- **Documento de estándares de desarrollo:** aquí se plasmarán las metodologías y estándares de desarrollo que se usarán.

Proceso de Elaboración (Análisis y diseño)

Para la fase de elaboración se deben tener contemplados los siguientes entregables:

- **Documento de especificación de requerimientos de software.** este documento contiene todos los casos de uso y su especificación, muestra los actores que participan en el sistema, diagramas de casos de uso, escenarios, así como las condiciones del sistema.
- **Documento de arquitectura de software.** este documento contiene la arquitectura del proyecto.
- **Lista de riesgos.** en este documento es la lista de riesgos actualizada generada en la fase de inicio.
- **Generación de un prototipo.** El propósito de generar un prototipo en esta fase es para ofrecer ayuda en la transición entre diseño y requerimientos, la reutilización de los componentes y ofrecer una demostración a los clientes de cómo funciona el sistema administrado.

Proceso de Construcción (Implementación)

Para la fase de construcción se deben tener contemplados los siguientes entregables:

- **Documento de orden de trabajo de ingeniería de software** este documento describe las actividades que han de ser realizadas así como las salidas esperadas, se hace referencia a la descripción de actividades a ejecutar así como los documentos o productos (código) que serán producidos.
- **Documento de evaluación del estatus del proyecto.** este documento contiene un reporte del progreso del trabajo, resultados de hitos completados y una lista de acciones para corregir cualquier desviación de la ejecución del proyecto.
- **Completar el análisis, diseño, desarrollo y pruebas.** el propósito de generar todas estas tareas es el de completar toda la funcionalidad requerida del sistema en cuestión.
- **Generación de versiones.** con esta actividad se pretende generar versiones del software, alpha, beta u otras pruebas de liberación, cada vez más estables.
- **Documento de orden de trabajo de ingeniería de software**
- **Documento de evaluación del estatus del proyecto.**

Proceso de pruebas

Para la fase de pruebas se deben tener contemplados los siguientes entregables:

- **Matriz de pruebas:** documento con todos los casos de pruebas contemplados para el sistema.
- **Documento de evidencias de prueba:** documento en que se deberá evidencia el correcto funcionamiento del sistema según las especificaciones planteadas

Proceso de Transición (puesta en marcha)

Para la fase de transición se deben tener contemplados los siguientes entregables:

- **Producto.** el producto es el propósito. El esfuerzo del proyecto entero acoplado para crear un producto que provea un beneficio al cliente.
- **Material de soporte para el usuario final (documentación)** el material que ayuda al usuario final a aprender, usar, operar, y mantener el producto, debe ser completado de acuerdo con los requerimientos.

Iteraciones

Se realizaran iteraciones para poder generar el producto final:

- **Iteración 1:** el proceso incluir todos los módulos funcionales del sistema con todas fases del desarrollo se entregará un producto funcional y útil, después de entregar cada prototipo la federación deberá generar un documento con observaciones del mismo que serán ejecutadas en la siguiente fase(los errores funcionales se resolverán en esta fase):
- **Iteración 2:** en esta iteración se realizarán todos los cambios en el sistema y modificaciones que sean necesarias para cumplir con las especificaciones finales de la federación.
- **Iteración 3:** solo de ser necesario se realizaría una tercera iteración que será planeada al final de la segunda.

Tiempos	Resultados	Descripción	Fase	Iteración
4 semanas	• Solicitudes del cliente. • Visión del proyecto de software • Plan de desarrollo de software. • Lista de riesgos. • Plan de iteración • Documento Análisis y determinación de requerimientos	Levantamiento de todos los requisitos del sistema en esta etapa se deberán programar múltiples reuniones con el cliente y definir alcances y limitantes, tanto como riesgos en el desarrollo	Fase de inicio	**1**
5 semanas	• Documento de especificación de requerimientos de software • Documento de arquitectura • Lista de riesgos • Generación de un prototipo.	En esta etapa se realizará el análisis y diseño de la solución también se deberá generar un prototipo para que el cliente pueda retroalimentar de mejor manera en esta etapa.	Fase de Elaboración	

10 semanas	• Documento de orden de trabajo de ingeniería de software • Documento de evaluación del estatus del proyecto. • Generación de versiones se tendrán las siguientes versiones : • Alpha: módulo de gestión de usuarios y seguridad , y plantillas de diseño para el sistema • Beta: módulo de registro integrado • Gamma: módulo de gestión de competencias integrado • Delta: módulo Financiero integrado • Icarus: Aplicación móvil Cada versión debe ser presentada y revisada por la federación	Desarrollo de software y pruebas unitarias de los mismos	Fase de Construcción	
1 semana	• Matriz de pruebas	Pruebas integradas de todos los módulos	Fase de pruebas	
2 semanas	• Producto • Documentación	Puesta en producción del producto	Fase de Transición	
2 semana	Actualizaciones de los siguientes documentos • Solicitudes del cliente. • Visión del proyecto de software	Después de entregar la versión delta del producto se deberá reunir con el cliente y evaluar posibles cambios en el producto	Fase de inicio	**2**

	• Plan de desarrollo de software. • Lista de riesgos. • Plan de iteración • Requerimientos técnicos			
2 semanas	• Documento de control de cambios	En esta fase se harán el análisis y diseño de los cambios que sufra el producto	Fase de Elaboración	
4 semanas	Generación de la versión épsilon(Final) del proyecto que deberá contener todos	Se elaborara una nueva versión del producto en base a los cambios solicitados	Fase de Construcción	
1 semanas	• Matriz de pruebas actualizadas • Documento de evidencias de prueba	Pruebas de la versión final del producto	Fase de pruebas	
2 semanas	• Producto • Documentación Final	Puesta en producción del producto final	Fase de Transición	

2.3. Requerimientos del Sistema

2.3.1. Requerimientos Funcionales

Módulo de Perfiles Personales

El Módulo debe poder permitir gestionar atletas, entrenadores, clubes, escuelas, miembros de junta directiva y jueces

Identificación del requerimiento:	PER01
Nombre del Requerimiento:	Gestión de Atletas
Características:	El sistema debe poder (Ingresar, Actualizar) Atletas
Descripción del requerimiento:	Almacenar Datos generales del surfista, Datos académicos, Datos técnicos, Patrocinadores, logros obtenidos, así como también asignarlos a las escuelas y clubes deportivos.
Prioridad del requerimiento: Alta	

Identificación del requerimiento:	PER02
Nombre del Requerimiento:	Gestión de Entrenadores
Características:	El sistema debe poder (Ingresar, Actualizar) Entrenadores
Descripción del requerimiento:	Almacenar Datos generales de entrenador
Prioridad del requerimiento: Alta	

Identificación del requerimiento:	PER03
Nombre del Requerimiento:	Gestión de Clubes
Características:	El sistema debe poder (Ingresar, Actualizar) Clubes
Descripción del requerimiento:	Almacenar Datos que refieren al club deportivo y asignar Entrenador
Prioridad del requerimiento: Alta	

Identificación del requerimiento:	PER04
Nombre del Requerimiento:	Gestión de Escuelas Deportivas
Características:	El sistema debe poder (Ingresar, Actualizar) Escuelas
Descripción del requerimiento:	Almacenar Datos que refieren a la escuela y asignar Entrenadores
Prioridad del requerimiento: Alta	

Identificación del requerimiento:	PER05
Nombre del Requerimiento:	Gestión de miembros de junta directiva
Características:	El sistema debe poder (Ingresar, Actualizar) miembros de junta directiva
Descripción del requerimiento:	Almacenar Datos generales de los miembros
Prioridad del requerimiento: Alta	

Identificación del requerimiento:	PER06
Nombre del Requerimiento:	Gestión de Jueces
Características:	El sistema debe poder (Ingresar, Actualizar) Jueces
Descripción del requerimiento:	Almacenar Datos generales de los Jueces involucrados en las competencias
Prioridad del requerimiento: Alta	

Módulo de Seguridad

Debe permitir gestionar las cuentas de usuarios autorizados para ingresar al sistema, administrar niveles de acceso por tipo de usuario.

Identificación del requerimiento:	SEG01
Nombre del Requerimiento:	Gestión de usuarios
Características:	El sistema debe poder administrar las cuentas de usuarios (Ingresar, Actualizar), que les permitan autenticarse para ingresar al sistema
Descripción del requerimiento:	El Administrador podrá gestionar usuarios almacenando datos personales, funciones o roles que desempeña, perfil de usuario que necesita y contraseñas
Prioridad del requerimiento: Alta	

Identificación del requerimiento:	SEG02
Nombre del Requerimiento:	Asignación de permisos de acceso por medio de roles
Características:	El sistema debe poder manejar asignación roles a los usuarios
Descripción del requerimiento:	El Administrador podrá asignar perfiles de usuarios requeridos (Administradores, usuarios estándar, juez) que contienen los permisos necesarios y niveles de acceso.
Prioridad del requerimiento: Alta	

Módulo de Ingresos y Gastos.

El módulo debe permitir ingresar y mostrar los gastos y los ingresos utilizados por la FESASURF, informando como son utilizados los recursos.

Identificación del requerimiento:	IG01
Nombre del Requerimiento:	Gestionar las cuentas involucradas.
Características:	El sistema debe gestionar las cuentas bancarias necesarias
Descripción del requerimiento:	Administrar las cuentas de ingresos y egresos requeridas
Prioridad del requerimiento: Alta	

Identificación del requerimiento:	IG02
Nombre del Requerimiento:	Registro de Fondos
Características:	El sistema debe llevar un registro de donde se invirtieron los fondos captados
Descripción del requerimiento:	Registrar los destinos finales de los fondos, en que fueron invertidos.
Prioridad del requerimiento: Alta	

Identificación del requerimiento:	IG03
Nombre del Requerimiento:	Registrar los orígenes de los fondos
Características:	El sistema debe tener un registro de quien o quienes son las personas, entidades que dan apoyo financiero
Descripción del requerimiento:	Registrar los fondos captados de INDES o patrocinadores, además recopilar información histórica de las aportaciones recibidas por los patrocinadores.
Prioridad del requerimiento: Alta	

Identificación del requerimiento:	IG04
Nombre del Requerimiento:	Registrar Transacciones
Características:	El sistema debe tener la funcionalidad de hacer registros tanto de ingresos como de gastos
Descripción del requerimiento:	Registrar los ingresos tanto de INDES como patrocinadores y los gastos incurridos
Prioridad del requerimiento: Alta	

Identificación del requerimiento:	IG05
Nombre del Requerimiento:	Consultar Saldos
Características:	El sistema debe tener la funcionalidad de hacer consultas de las cuentas bancarias y ver el detalle de ingresos como de gastos en esa cuenta
Descripción del requerimiento:	Obtener estados de cuentas tanto de ingresos, gastos y transacciones en las mismas
Prioridad del requerimiento: Alta	

Módulo de Competencias.

Debe permitir gestionar cada una de las actividades de las competencias (circuitos) que se realizan durante el año, así también dar soporte a para cada uno de los siguientes elementos:

- Gestión de Circuitos (Crear, Modificar)
- Gestión de Categorías (Agregar, Modificar)
- Gestión de Atletas Participantes (Inscribir, Desinscribir)
- Gestión de Rondas (Crear, Modificar)
- Gestión de Heats para cada Ronda (Generar, Modificar)
- Gestión de Jueces Participantes (Agregar)
- Gestión de Jueces para cada Heat (Asignar)
- Gestión de Atletas para cada Heat (Asignar)
- Gestión puntuaciones (Agregar, Modificar)
- Consultar Ranking
- Calculo de puntuaciones por cada Heat
- Calculo de puntuaciones por cada ronda
- Calculo de puntuaciones por cada circuito
- Calculo de puntuaciones por cada categoría
- Consolidación de puntajes finales para ranking general

Identificación del requerimiento:	COM01
Nombre del Requerimiento:	Gestión de Circuitos
Características:	El sistema debe poder (Crear, Modificar) Circuitos
Descripción del requerimiento:	Para poder gestionar Circuitos el sistema debe tener en cuenta que los circuitos incluyen tanto rondas, Heats, puntuaciones así como también atletas y jueces participantes en cada una de ellas
Prioridad del requerimiento: Alta	

Identificación del requerimiento:	COM02
Nombre del Requerimiento:	Gestión de Categorías
Características:	El sistema debe poder (Agregar, Modificar) Categorías
Descripción del requerimiento:	Almacenar Datos de las categorías se espera que estos datos sean prácticamente fijos en el sistema ya que la FESASURF tiene categorías definidas con sus características para las competencias (Júnior, Longboard, Damas, Bodyboard, Master, Open, Groms)
Prioridad del requerimiento: Alta	

Identificación del requerimiento:	COM03
Nombre del Requerimiento:	Gestión de atletas participantes
Características:	El sistema permitirá la inscripción y descripción de atletas en competencias
Descripción del requerimiento:	Se deberá crear una vista a manera de tabla que permita agregar participante a las competencia también deberá poder eliminar participantes del mismo a manera de Desinscribir
Prioridad del requerimiento: Alta	

Identificación del requerimiento:	COM04
Nombre del Requerimiento:	Gestión de Rondas
Características:	El sistema debe poder (Crear, Modificar) Rondas
Descripción del requerimiento:	Se debe almacenarla información de cada una de las rondas que se den en cada circuito, estas dependerán de la cantidad de surfistas registrados en el mismo.
Prioridad del requerimiento: Alta	

Identificación del requerimiento:	COM05
Nombre del Requerimiento:	Gestión de Heat para cada ronda
Características:	El sistema debe poder (Generar, Modificar) Heats que tengan las rondas
Descripción del requerimiento:	Se debe de almacenar la información de cada uno de los Heats que se den por ronda, para cada Heat se asignara automáticamente la cantidad de surfistas, teniendo como mínimo tres surfistas y en casos excepcionales hasta 5
Prioridad del requerimiento: Alta	

Identificación del requerimiento:	COM06
Nombre del Requerimiento:	Gestión jueces participantes
Características:	El sistema permitirá asignar jueces a las competencias
Descripción del requerimiento:	Se deberá crear una vista a manera de tabla que permita agregar Los jueces a la competencia
Prioridad del requerimiento: Alta	

Identificación del requerimiento:	COM07
Nombre del Requerimiento:	Gestión de Jueces y Atletas para cada Heat
Características:	El sistema debe Gestionar los atletas y jueces correspondientes que tengan los Heats
Descripción del requerimiento:	Siempre que se realice un nuevo Heat se debe tener en cuenta que atletas y jueces incluye ese Heat del total de todos los jueces y atletas participantes, cabe mencionar que un Heat tiene únicamente 10 olas
Prioridad del requerimiento: Alta	

Identificación del requerimiento:	COM08
Nombre del Requerimiento:	Gestión de puntuaciones
Características:	El sistema debe poder (Agregar, Modificar) Puntaciones asociadas a las rondas y a los Heats de un surfista
Descripción del requerimiento:	Se deben poder guardar puntaciones de todos los surfistas que participen en competencia y se debe poder visualizar y filtrar estas puntuaciones, los filtros pueden ser por competencia(circuito), por ronda, por Heat, por Surfista, por categoría, por fecha y deben poder ordenarse por el valor de la puntuación
Prioridad del requerimiento: Alta	

Identificación del requerimiento:	COM09
Nombre del Requerimiento:	Consultar de Ranking
Características:	El sistema mostrar el ranking de los surfistas nacionales
Descripción del requerimiento:	Se deberá crear un algoritmo que permita el cálculo del ranking global de surfistas, y mostrarse ordenadamente en una página web
Prioridad del requerimiento: Alta	

Identificación del requerimiento:	COM10
Nombre del Requerimiento:	Asignaciones de Puntos para el Ranking
Características:	El sistema debe asignar los puntos al ranking
Descripción del requerimiento:	Siempre que se realice el cálculo de puntajes en un Heat exitosamente se debe actualizar automáticamente los puntos de la ronda a la que pertenece el Heat y de igual manera el ranking hasta el momento para poder apreciar las posiciones de los atletas participantes
Prioridad del requerimiento: Alta	

Módulo de Noticias

El Módulo debe poder permitir gestionar las noticias recientes de la FESASURF.

Identificación del requerimiento:	NT01
Nombre del Requerimiento:	Gestión de Noticias
Características:	El sistema debe poder (Ingresar, Actualizar) Noticias
Descripción del requerimiento:	Almacenar Datos generales de las Noticias recientes que se darán a conocer
Prioridad del requerimiento: Media	

Aplicación Móvil.

Debe permitir la consulta de las competencias y de las noticias de la federación mediante una App.

Identificación del requerimiento:	MOV01
Nombre del Requerimiento:	Notificaciones de usuario
Características:	La aplicación móvil podrá realizar notificaciones al usuario
Descripción del requerimiento:	Notificar al usuario acerca de las actividades relacionadas a la FESASURF y noticias importantes del surf.
Prioridad del requerimiento: Alta	

Identificación del requerimiento:	MOV02
Nombre del Requerimiento:	Consulta de Información
Características:	La aplicación móvil podrá permitir al usuario buscar información
Descripción del requerimiento:	El usuario podrá consultar información de su interés acerca de competencias y resultados de las mismas, noticias, calendarios de actividades.
Prioridad del requerimiento: Alta	

2.3.2. Requerimientos No Funcionales

Identificación del requerimiento:	RNF01
Nombre del Requerimiento:	Diseño de la interfaz del sistema
Características:	El sistema deberá de tener una interfaz de usuario alusiva a la FESASURF
Descripción del requerimiento:	En general La interfaz de usuario debe ajustarse a las características de FESASURF y el ámbito de sus actividades además de ser intuitiva y fácil de usar.
Prioridad del requerimiento: Alta	

Identificación del requerimiento:	RNF02
Nombre del Requerimiento:	Nivel de Usuario
Características:	Garantizara al usuario el acceso de información de acuerdo al nivel que posee.
Descripción del requerimiento:	Acceso de información restringido
Prioridad del requerimiento: Alta	

Identificación del requerimiento:	RNF03
Nombre del Requerimiento:	Seguridad en información
Características:	El sistema garantizara a los usuarios una seguridad en cuanto a la información que se posee en el sistema.
Descripción del requerimiento:	Garantizar la seguridad del sistema con respecto a la información y datos que se manejan
Prioridad del requerimiento: Alta	

Identificación del requerimiento:	RNF04
Nombre del Requerimiento:	Administrar el tiempo de sesión del usuario
Características:	El sistema debe manejar tiempos de sesión
Descripción del requerimiento:	Los tiempos de sesión deben estar incluidos en el sistema gestionando inactividad por parte del usuario en un periodo de tiempo
Prioridad del requerimiento: Media	

Identificación del requerimiento:	RNF05
Nombre del Requerimiento:	Cambios Regulares de Contraseña
Características:	El sistema debe tener en cuenta los cambios regulares de contraseña
Descripción del requerimiento:	Los cambios regulares de contraseña deben ser incluidos por motivos de seguridad y cada cierto tiempo deben ser modificadas
Prioridad del requerimiento: Media	

Identificación del requerimiento:	RNF06
Nombre del Requerimiento:	Bloqueos de Cuenta
Características:	El sistema debe bloquear cuentas al ingresar credenciales erróneas
Descripción del requerimiento:	Se debe bloquear la cuenta de los usuarios que al ingresar mal la contraseña un numero definido de veces no hayan podido entrar al sistema
Prioridad del requerimiento: Media	

Identificación del requerimiento:	RNF07
Nombre del Requerimiento:	Texto/Mensajes parametrizables.
Características:	
Descripción del requerimiento:	El sistema debe proveer una interfaz para poder configurar todos los texto/Mensaje que se muestren de cara al usuario, para los servicios expuestos y procesos que interactúan con el número.
Prioridad del requerimiento: Alta	

Identificación del requerimiento:	RNF08
Nombre del Requerimiento:	Rendimiento
Características:	Todos los procesos deben estar preparados para soportar casos masivos.
Descripción del requerimiento:	Toda funcionalidad del sistema y transacción debe responder al usuario en menos de 4 segundos.
Prioridad del requerimiento: Alta	

Identificación del requerimiento:	RNF09
Nombre del Requerimiento:	Disponibilidad
Características:	Disponibilidad inmediata del sistema en las competencias
Descripción del requerimiento:	El sistema debe tener una disponibilidad del 99,99% de las veces en que un usuario intente accederlo.
Prioridad del requerimiento: Alta	

Identificación del requerimiento:	RNF10
Nombre del Requerimiento:	Mantenibilidad
Características:	Manuales del Sistema
Descripción del requerimiento:	El sistema debe disponer de manuales (usuario, instalación, técnico) que permita realizar operaciones de mantenimiento con el menor esfuerzo posible.
Prioridad del requerimiento: Media	

2.4. Estándares de Diagramas

Para la elaboración de diagramas se utilizará El lenguaje unificado de modelado (UML 2.4), el cual es un estándar de modelado de sistemas de software.

2.4.1. Casos de Uso

Convención de identificadores

Elemento	Descripción	Ejemplo
Identificador de Requerimiento del caso de uso	Se utilizará un identificador para relacionar el requerimiento con la descripción del caso de uso, este se compondrá de un código que inicie con un distintivo del módulo como “PER”, “IG”, “COM” para hacer referencia al módulo de perfiles personales, competencias etc. Seguido de un correlativo.	PER01 IG03 COM04

Plantilla para descripción de casos de Uso

Identificación del requerimiento:		
Nombre CU:		
Objetivo:		
Descripción.		
Precondición.		
Flujo de Éxito.		

Flujo Alternativo.		
Pos-condición		
Puntos de Ampliación		
Frecuencia.		
Prioridad del CU:		

2.4.2. Diagramas de Secuencia

Convención de nombres

Elemento	Descripción	Ejemplo
Nombre del actor	Para el nombre del autor involucrado se utilizará la notación CamelCase en su variante UpperCamelCase.	
Mensaje	Para el nombre de los mensajes se utilizará la notación CamelCase en su variante UpperCamelCase.	
Instancia	Para el nombre de la Instancia involucrada se utilizará la notación CamelCase en su variante UpperCamelCase	

2.4.3. Modelo de Dominio

Convención de nombres

Elemento	Descripción	Ejemplo
Nombre de Clase conceptual	Para el nombre de las clases se utilizará la notación UpperCamelCase.	-Club -Entrenador -Patrocinador
Atributo de clase conceptual	Para el nombre de los atributos se utilizará la notación UpperCamelCase.	-NombreCategoria -EdadMin -EdadMax
Nombre de relación	Para el nombre de la relación entre clases de dominio se utilizará la notación UpperCamelCase.	

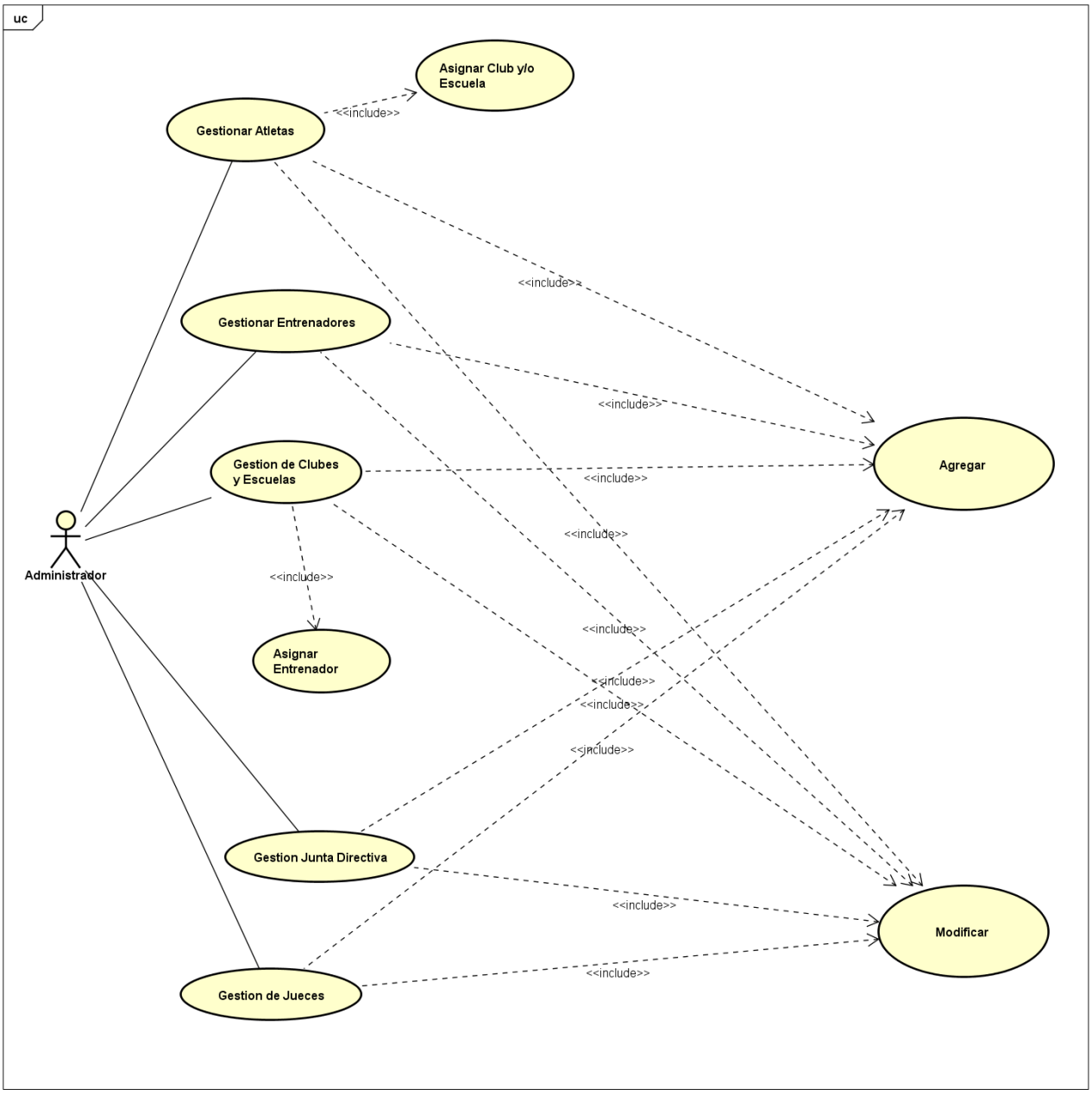
2.4.4. Diagramas de Clases

Convención de nombres

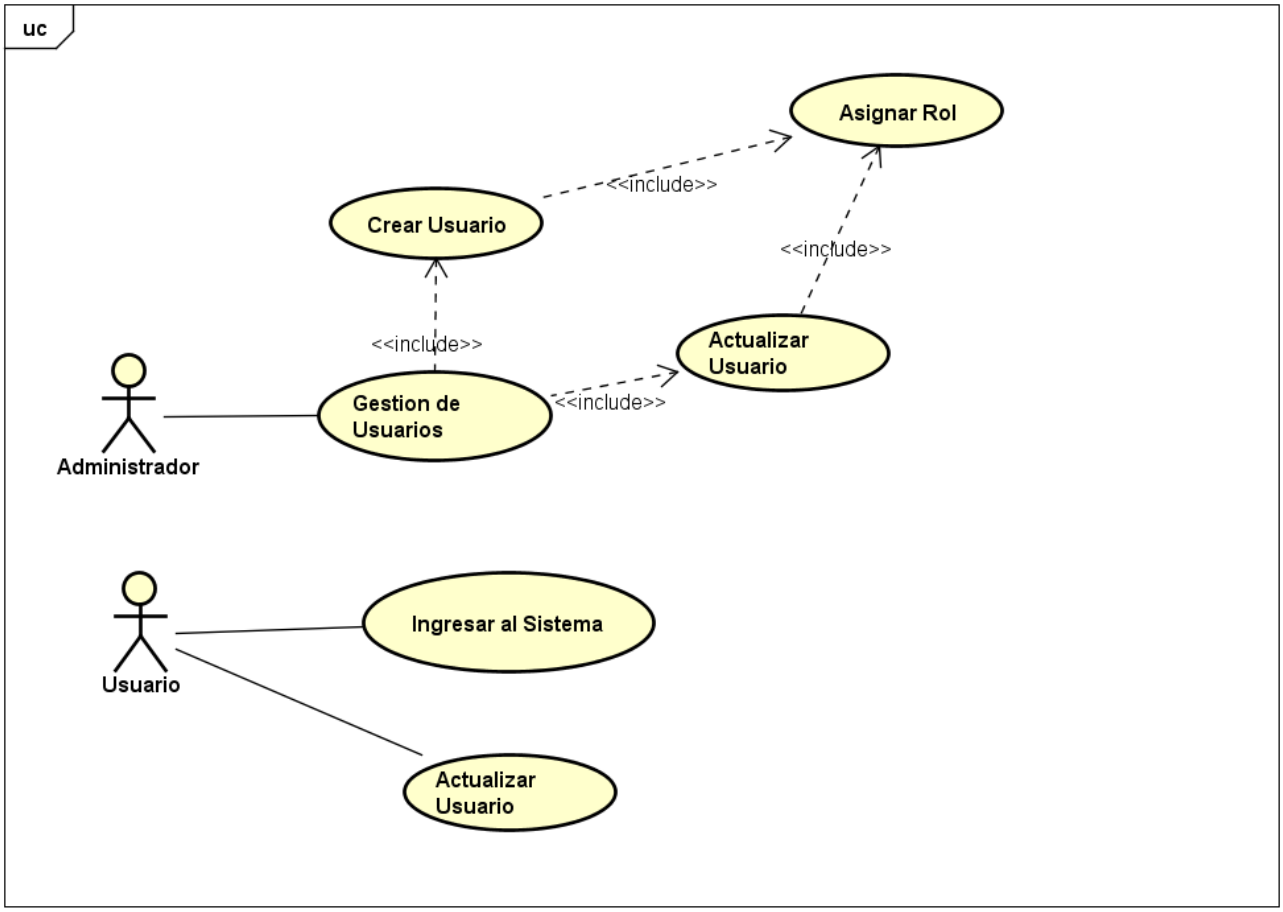
Elemento	**Descripción**	**Ejemplo**
Nombre de Clase	Para el nombre de Clase, se utilizará la notación UpperCamelCase.	
Atributo de Clase	Para el nombre de atributo, se utilizará la notación UpperCamelCase.	
Método de Clase	Para el nombre de método, se utilizará la notación UpperCamelCase. El nombre del método deberá comenzar con un verbo en infinitivo.	-CrearRol () -ActualizarUsuario ()
Parámetro de métodos de Clase	Para nombrar los parámetros que serán utilizados en los métodos, se utilizará la notación UpperCamelCase	-IdCircuito -NumAtletasRonda

2.5. Diagramas de Caso de Uso general del Sistema

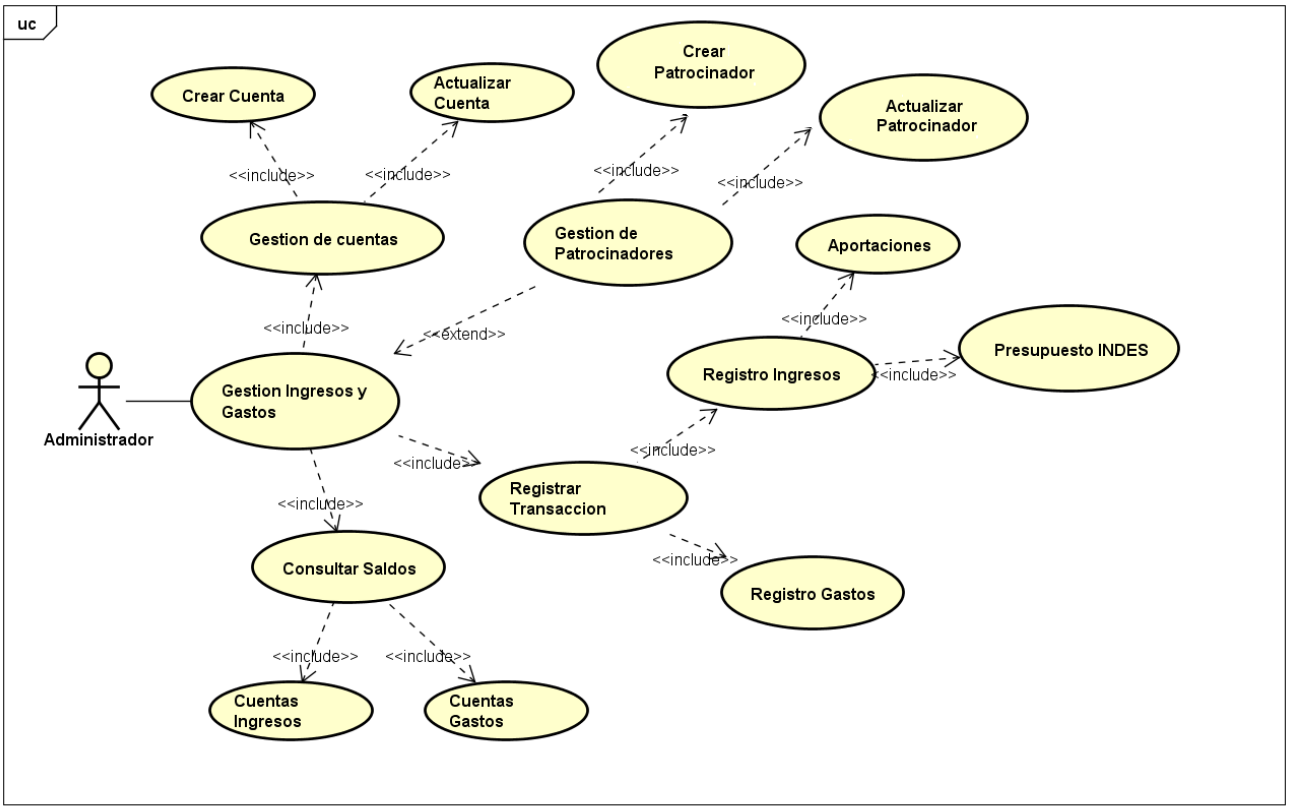
Módulo de Perfiles Personales



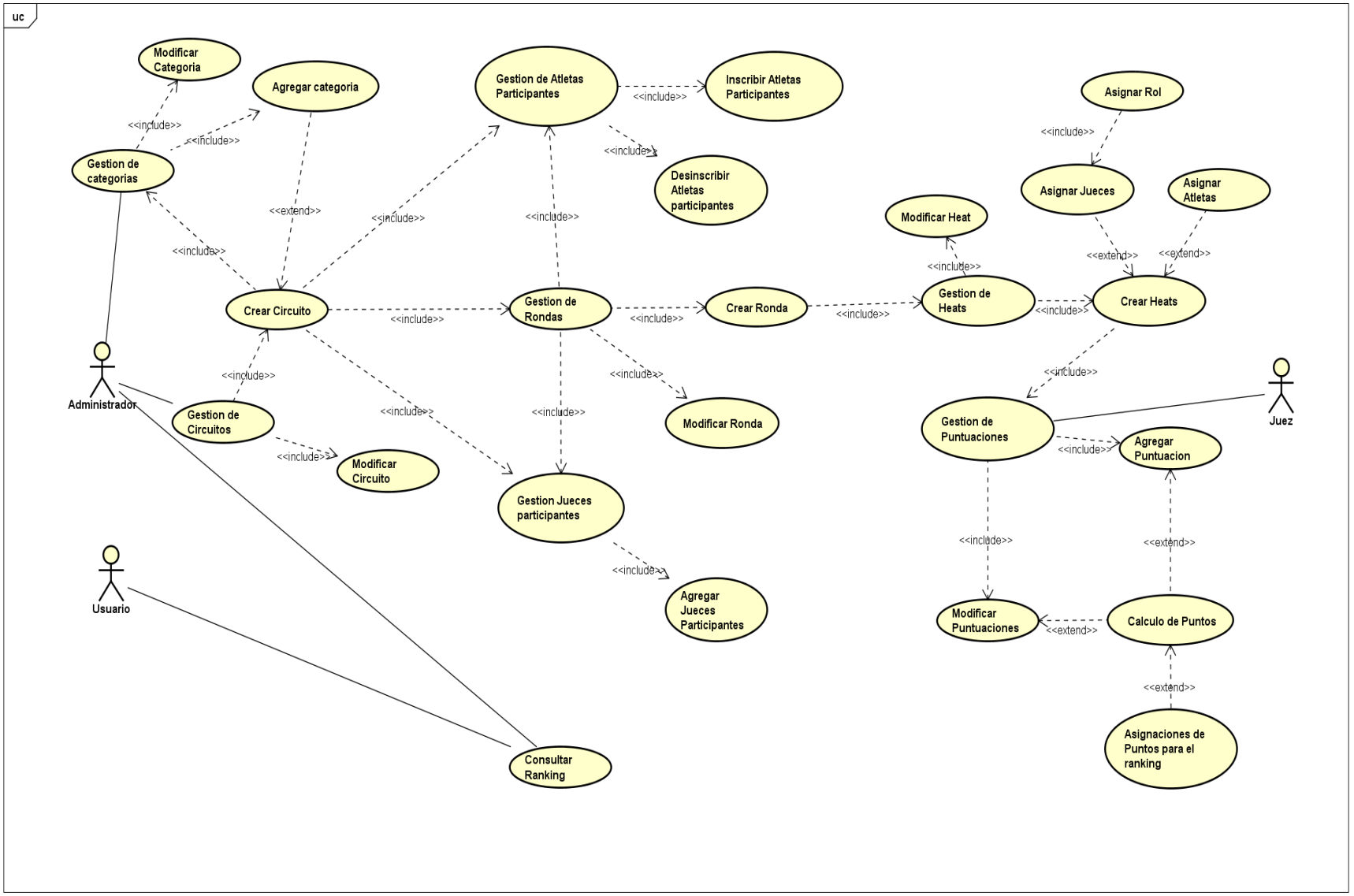
Módulo de Seguridad



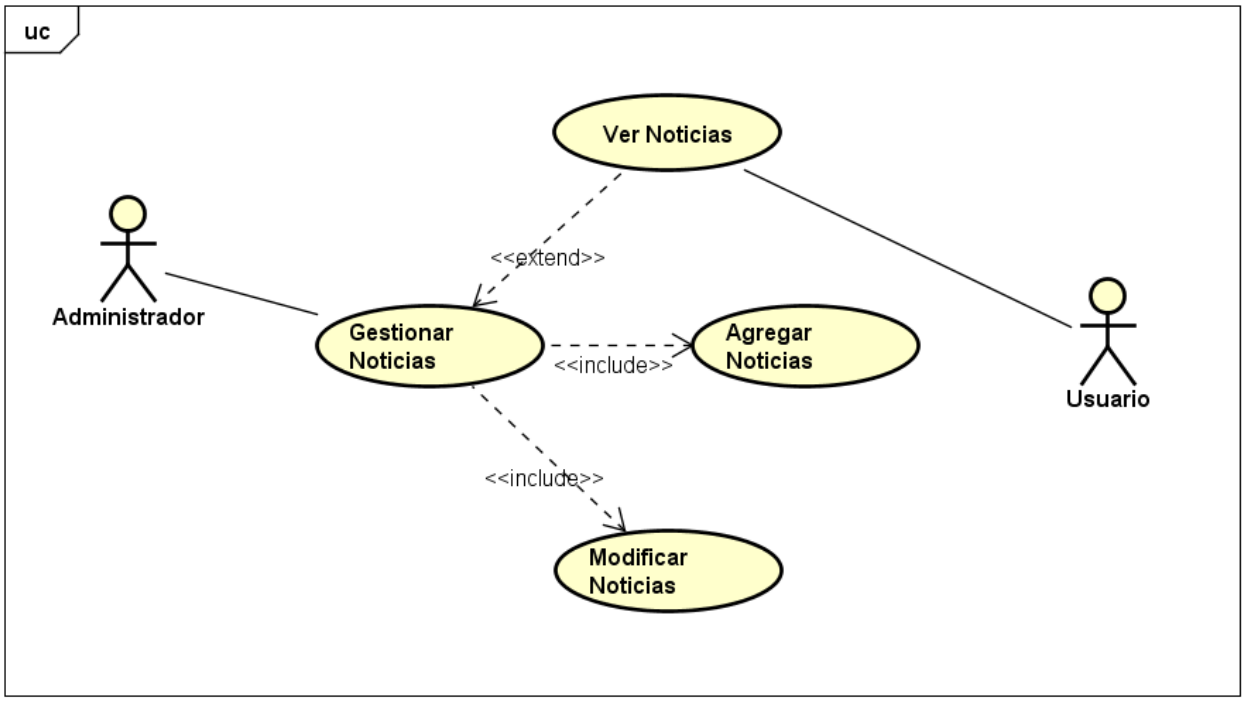
Módulo de Ingresos y Gastos.



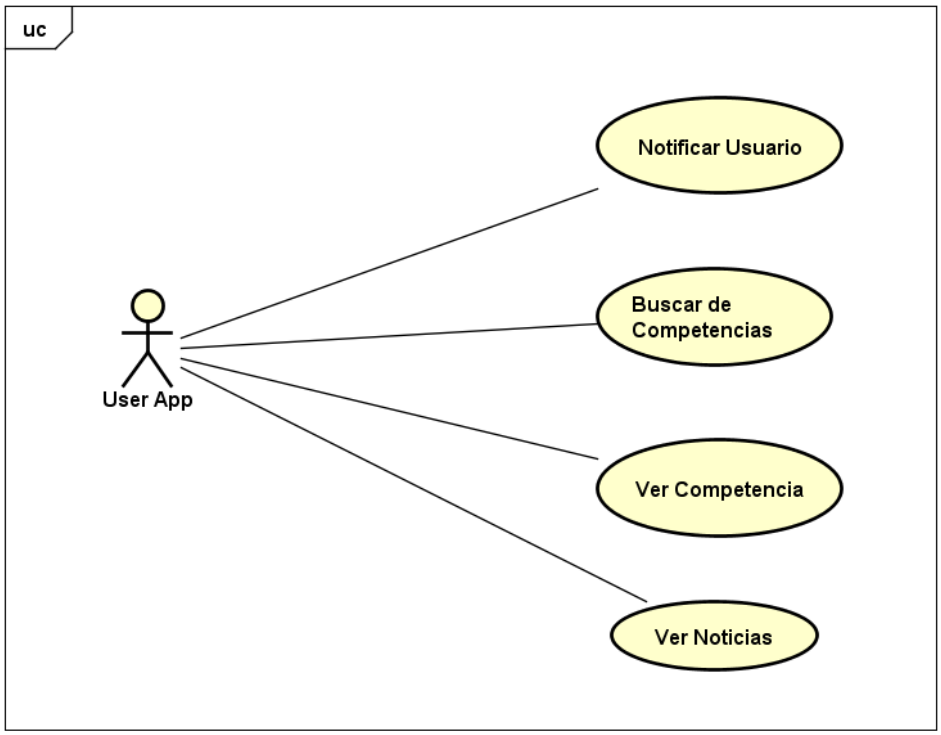
Módulo de Competencias.



Módulo de Noticias



Aplicación Móvil.



2.5.1. Descripción de casos de uso

Módulo de Perfiles Personales

Identificación del requerimiento:	PER01	
Nombre CU:	Agregar Atletas	
Objetivo:	Agregar los datos de los nuevos atletas involucrados en las actividades FESASURF.	
Descripción.	El Administrador del sistema podrá agregar nuevos atletas	
Precondición.	El administrador del sistema debe de haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema”	
Flujo de Éxito.	Paso	Acción
	1	El administrador del sistema ingresa al módulo de perfiles personales y elige la opción gestión de atletas.
	2	Selecciona la opción Agregar Atleta
	3	Llena los datos generales de la persona
	4	Se llena los datos adicionales requeridos para el registro de los atletas, Datos académicos, Datos técnicos, Patrocinadores de los que ha recibido apoyo, logros obtenidos (Ver Anexos- Formulario1) y confirma la operación al sistema en “Agregar”
	5	El sistema realiza una validación de los datos del atleta y los almacena
Flujo Alternativo.	Paso	Acción
	5	Si los datos tienen inconsistencias, se solicita la corrección, se valida y los almacena
Pos-condición	El nuevo atleta es agregado exitosamente y listo para utilizarse en el módulo de competencias	
Puntos de Ampliación	“Asignar Escuela y/o Club”	
Frecuencia.	Cada vez que sea necesario agregar un nuevo atleta	
Prioridad del CU:	Alta	

Identificación del requerimiento:	PER02	
Nombre CU:	Agregar Entrenadores	
Objetivo:	Agregar los datos de los nuevos entrenadores involucrados en las actividades FESASURF.	
Descripción.	El Administrador del sistema podrá agregar Entrenadores	
Precondición.	El administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema”	
Flujo de Éxito.	Paso	Acción
	1	El administrador del sistema ingresa al módulo de perfiles personales y elige la opción gestionar entrenadores
	2	Selecciona la opción Agregar Entrenador
	3	Llena los datos generales del entrenador (Ver Anexos-Formulario1) y confirma la operación al sistema en “Agregar”
	4	El sistema realiza una validación de los datos del entrenador y los almacena
Flujo Alternativo.	Paso	Acción
	4	Si los datos tienen inconsistencias, se solicita la corrección, se valida y los almacena
Pos-condición	El nuevo entrenador es agregado exitosamente y listo para utilizarse	
Frecuencia.	Cada vez que sea necesario agregar un nuevo entrenador	
Prioridad del CU:	Alta	

Identificación del requerimiento:	PER03	
Nombre CU:	Agregar Clubes y Escuelas	
Objetivo:	Agregar los datos de los nuevos clubes y escuelas involucrados en las actividades FESASURF.	
Descripción.	El Administrador del sistema podrá crear clubes y escuelas	
Precondición.	El Administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema”	
Flujo de Éxito.	Paso	Acción
	1	El administrador del sistema ingresa al módulo de perfiles personales y elige la opción gestión de Clubes y Escuelas
	2	Selecciona la opción Agregar Nuevo
	3	Llena los datos generales (Dirección, Teléfono, Correo, Encargado, Fecha de Fundación, tipo si es Escuela o Club y el Nombre de esta) luego confirma la operación al sistema en “Agregar”
	4	El sistema realiza una validación de los datos del club o escuela y los almacena
Flujo Alternativo.	Paso	Acción
	4	Si los datos tienen inconsistencias, se solicita la corrección, se valida y los almacena
Puntos de Ampliación	“Asignar Entrenador”	
Pos-condición	El nuevo club o escuela es agregado exitosamente y listo para utilizarse.	
Frecuencia.	Cada vez que sea necesario crear un nuevo club o escuela	
Prioridad del CU:	Alta	

Identificación del requerimiento:	PER04	
Nombre CU:	Asignar Entrenador	
Objetivo:	Definir los entrenadores de las escuelas y clubes	
Descripción.	El administrador del sistema podrá asignar los entrenadores a las escuelas y clubes	
Precondición.	El administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema”	
Flujo de Éxito.	Paso	Acción
	1	El administrador del sistema ingresa al módulo de perfiles personales y selecciona la opción Gestionar Clubes y Escuelas
	2	Elige el club o escuela y selecciona Asignar Entrenador
	3	El administrador asigna el entrenador para la escuela o club (en base a la información proporcionada por la escuela o club) y luego selecciona Asignar
	4	El sistema asigna el Entrenador seleccionado a la Escuela o club
Pos-condición	El Entrenador ha sido asignado a la Escuela o club	
Frecuencia.	Cada vez que se necesite Asignar un nuevo Entrenador a una Escuela o club	
Prioridad del CU:	Alta	

Identificación del requerimiento:	PER05	
Nombre CU:	Agregar nuevos miembros de junta directiva	
Objetivo:	Agregar los datos de los nuevos miembros de junta directiva involucrados en las actividades FESASURF.	
Descripción.	El Administrador del sistema podrá crear perfiles de miembros de junta directiva	
Precondición.	El administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema”	
Flujo de Éxito.	Paso	Acción
	1	El administrador del sistema ingresa al módulo de perfiles personales y elige la opción gestión de miembros de Junta
	2	Selecciona la opción Agregar Miembro
	3	Llena los datos generales del miembro de Junta Directiva (Ver Anexos- Formulario1) y confirma la operación al sistema en “Agregar”
	4	El sistema realiza una validación de los datos del miembro de junta directiva a agregar y los almacena
Flujo Alternativo.	Paso	Acción
	4	Si los datos tienen inconsistencias, se solicita la corrección, se valida y los almacena
Pos-condición	El nuevo miembro de junta es agregado exitosamente y listo para utilizarse	
Frecuencia.	Cada vez que sea necesario agregar un nuevo miembro de junta directiva	
Prioridad del CU:	Alta	

Identificación del requerimiento:	PER06	
Nombre CU:	Agregar Jueces	
Objetivo:	Agregar los datos de los nuevos Jueces involucrados en las actividades FESASURF.	
Descripción.	El Administrador del sistema podrá agregar jueces	
Precondición.	El Administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema”	
Flujo de Éxito	Paso	Acción
	1	El administrador del sistema ingresa al módulo de perfiles personales y elige la opción gestión de jueces
	2	Selecciona la opción Agregar Juez
	3	Llena los datos generales del juez involucrado en la competencia (Ver Anexos- Formulario1) y confirma la operación al sistema en “Agregar”
	4	El sistema realiza una validación de los datos del juez y los almacena
Flujo Alternativo.	Paso	Acción
	4	Si los datos tienen inconsistencias, se solicita la corrección, se valida y los almacena
Pos-condición	El nuevo juez es agregado exitosamente y listo para utilizarse en el módulo de competencias	
Frecuencia.	Cada vez que sea necesario agregar un nuevo juez	
Prioridad del CU:	Alta	

Identificación del requerimiento:	PER07	
Nombre CU:	Modificar Atletas	
Objetivo:	Realizar los cambios en los datos de los atletas involucrados en las actividades FESASURF.	
Descripción.	El Administrador del sistema podrá modificar los datos de los Atletas	
Precondición.	El administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema”	
Flujo de Éxito.	Paso	Acción
	1	El administrador del sistema ingresa al módulo de perfiles personales y elige la opción gestión de atletas.
	2	Elige el Atleta y Selecciona el icono de Modificar
	3	Modifica los datos generales de la Persona, entre ellos Datos académicos, Datos técnicos, Patrocinadores de los que ha recibido apoyo, logros obtenidos(Ver Anexos-Formulario1) y confirma la operación al sistema
	4	El sistema realiza una validación de los datos del atleta y los modifica
Flujo Alternativo.	Paso	Acción
	4	Si los datos tienen inconsistencias, se solicita la corrección, se valida y los modifica
Puntos de Ampliación	“Asignar Escuela y/o Club”	
Pos-condición	La información del atleta se ha modificado exitosamente y está lista para utilizarse en el módulo de competencias	
Frecuencia.	Cada vez que sea necesario modificar un perfil de atleta	
Prioridad del CU:	Alta	

Identificación del requerimiento:	PER08	
Nombre CU:	Modificar Entrenadores	
Objetivo:	Realizar los cambios en los datos de entrenadores involucrados en las actividades FESASURF.	
Descripción.	El administrador del sistema podrá modificar perfiles de Entrenadores	
Precondición.	El administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema”	
Flujo de Éxito.	Paso	Acción
	1	El administrador del sistema ingresa al módulo de perfiles personales y elige la opción gestionar entrenadores
	2	Elige el Entrenador y Selecciona el icono Modificar
	3	Modifica los datos generales del entrenador (Ver Anexos-Formulario1) y confirma la operación al sistema
	4	El sistema realiza una validación de los datos del entrenador y los modifica
Flujo Alternativo.	Paso	Acción
	4	Si los datos tienen inconsistencias, se solicita la corrección, se valida y los modifica
Pos-condición	El entrenador se ha modificado exitosamente y está listo para utilizarse	
Frecuencia.	Cada vez que sea necesario modificar un entrenador	
Prioridad del CU:	Alta	

Identificación del requerimiento:	PER09	
Nombre CU:	Modificar Clubes y Escuelas	
Objetivo:	Realizar los cambios en los datos de los clubes y escuelas involucrados en las actividades FESASURF.	
Descripción.	El administrador del sistema podrá modificar Clubes y Escuelas	
Precondición.	El administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema”	
Flujo de Éxito.	Paso	Acción
	1	El administrador del sistema ingresa al módulo de perfiles personales y elige la opción gestión de clubes y escuelas
	2	Elige el club o escuela y Selecciona el icono de Modificar
	3	Modifica los datos generales del clubo escuela (Dirección, Teléfono, Correo, Encargado, Fecha de Fundación, tipo si es Escuela o Club y el Nombre de esta) luego confirma la operación al sistema
	4	El sistema realiza una validación de los datos del club o escuela y los modifica
Flujo Alternativo.	Paso	Acción
	4	Si los datos tienen inconsistencias, se solicita la corrección, se valida y los modifica
Puntos de Ampliación	“Asignar Entrenador”	
Pos-condición	El club o escuela se ha modificado exitosamente y listo para utilizarse	
Frecuencia.	Cada vez que sea necesario modificar un club o escuela	
Prioridad del CU:	Alta	

Identificación del requerimiento:	PER10	
Nombre CU:	Modificar Miembros de Junta Directiva	
Objetivo:	Realizar los cambios en los datos de los miembros de junta directiva involucrados en las actividades FESASURF.	
Descripción.	El administrador del sistema podrá modificar perfiles de Miembros de Junta Directiva	
Precondición.	El administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema”	
Flujo de Éxito.	Paso	Acción
	1	El administrador del sistema ingresa al módulo de perfiles personales y elige la opción gestión de miembros de Junta
	2	Elige el miembro de junta directiva y Selecciona el icono Modificar
	3	Modifica los datos generales del miembro de Junta Directiva (Ver Anexos- Formulario1) y confirma la operación al sistema
	4	El sistema realiza una validación de los datos del miembro de junta directiva a editar y los modifica
Flujo Alternativo.	Paso	Acción
	4	Si los datos tienen inconsistencias, se solicita la corrección, se valida y los modifica
Pos-condición	El miembro de junta se ha modificado exitosamente y listo para utilizarse	
Frecuencia.	Cada vez que sea necesario modificar un miembro de junta	
Prioridad del CU:	Alta	

Identificación del requerimiento:	PER11	
Nombre CU:	Modificar Jueces	
Objetivo:	Realizar los cambios en los datos de Jueces involucrados en las actividades FESASURF.	
Descripción.	El administrador del sistema podrá modificar perfiles de Jueces	
Precondición.	El administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema”	
Flujo de Éxito.	Paso	Acción
	1	El administrador del sistema ingresa al módulo de perfiles personales y elige la opción gestión de jueces
	2	Elige el Juez y Selecciona el icono de Modificar
	3	Modifica los datos generales del juez involucrado en la competencia (Ver Anexos- Formulario1) y confirma la operación al sistema
	4	El sistema realiza una validación de los datos del juez y los modifica
Flujo Alternativo.	Paso	Acción
	4	Si los datos tienen inconsistencias, se solicita la corrección, se valida y los modifica
Pos-condición	El juez se ha modificado exitosamente y listo para utilizarse en el módulo de competencias	
Frecuencia.	Cada vez que sea necesario modificar un juez	
Prioridad del CU:	Alta	

Identificación del requerimiento:	PER12	
Nombre CU:	Asignar Escuela y/o Club	
Objetivo:	Definir las escuelas y clubes a las que pertenecen los atletas.	
Descripción.	El administrador del sistema podrá asignar los atletas a las escuelas y clubes correspondientes.	
Precondición.	El administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema”	
Flujo de Éxito.	Paso	Acción
	1	El administrador del sistema ingresa al módulo de perfiles personales y selecciona la opción Gestionar Atletas.
	2	Elige el atleta y selecciona Asignar Escuela y/o Club
	3	El administrador define la Escuela o Club a la que pertenece el atleta y luego selecciona Asignar
	4	El sistema Asigna el atleta a la Escuela o Club seleccionado
Flujo Alternativo.	Paso	Acción
Pos-condición	El Atleta ha sido asignado a la Escuela y/o Club	
Frecuencia.	Cada vez que se necesite Asignar un nuevo atleta a una Escuela y/o Club	
Prioridad del CU:	Alta	

Módulo de Seguridad

Identificación del requerimiento:	SEG01	
Nombre CU:	Crear Usuarios.	
Objetivo:	Definir las personas autorizadas para el uso del sistema.	
Descripción.	El administrador del sistema podrá crear el perfil de cada usuario con sus datos personales y el rol que desempeña en el sistema para el ingreso al mismo.	
Precondición.	El administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema” El usuario debe ser nuevo y no existir previamente en el sistema.	
Flujo de Éxito.	Paso	Acción
	1	El administrador del sistema ingresa al módulo de gestión de usuario y elige la opción nuevo usuario.
	2	Llena los datos (Nombres, Apellidos, Correo, Login, la Contraseña inicial, Rol)
	3	El sistema valida los datos, y guarda al nuevo usuario
Flujo Alternativo.	Paso	Acción
	3	Si los datos tienen inconsistencias, se solicita la corrección, se valida y se guarda al nuevo usuario.
Pos-condición	El usuario ha sido creado y está listo para ingresar y cambiar por primera vez su contraseña.	
Puntos de Ampliación	“Asignar Rol”	
Frecuencia.	Cada vez que sea necesario crear un nuevo usuario.	
Prioridad del CU:	Alta	

Identificación del requerimiento:	SEG02	
Nombre CU:	Actualizar Usuarios. (Vista del Administrador)	
Objetivo:	Modificar los atributos de los usuarios autorizados para el uso del sistema.	
Descripción.	El administrador del sistema podrá hacer cambios a las características de los usuarios del sistema	
Precondición.	El usuario a modificar debe existir previamente en el sistema. El administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema”	
Flujo de Éxito.	Paso	Acción
	1	El administrador del sistema ingresa al módulo de gestión de usuario y elige la opción modificar usuario.
	2	Elige al usuario a modificar.
	3	El sistema presenta los datos del perfil del usuario y habilita las opciones que puede cambiar un administrador
	4	El administrador puede: • Restablecer contraseña, • cambiar estado del usuario(Activo, Inactivo), • Cambiar Rol del usuario.
	5	El sistema valida y guarda los cambios.
Flujo Alternativo.	Paso	Acción
	5	Si los datos tienen inconsistencias, se solicita la corrección, se valida y se actualiza al usuario.
Pos-condición	El usuario ha sido modificado con nuevas propiedades.	
Puntos de Ampliación	“Asignar Rol”	
Frecuencia.	Cada vez que se requiera un cambio en un usuario.	
Prioridad del CU:	Alta	

Identificación del requerimiento:	SEG03	
Nombre CU:	Actualizar Usuarios. (Vista del Usuario)	
Objetivo:	Modificar las propiedades de los usuarios autorizados para el uso del sistema.	
Descripción.	Hacer cambios a las características de los usuarios del sistema	
Precondición.	El usuario del sistema debe existir previamente en el sistema. El usuario del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema”	
Flujo de Éxito.	Paso	Acción
	1	El usuario del sistema ingresa al módulo de gestión de usuario y elige la opción modificar su cuenta de usuario.
	2	El sistema presenta los datos del perfil del usuario (Nombres, Apellidos, Correo, Login, la Contraseña, Rol)
	3	El usuario puede: • Cambiar su contraseña, • Cambiar sus datos personales.
	4	El sistema valida y guarda los cambios.
Flujo Alternativo.	Paso	Acción
	4	Si los datos tienen inconsistencias, se solicita la corrección, se valida y se actualiza al usuario.
Pos-condición	El usuario del sistema ha modificado sus propiedades.	
Frecuencia.	Cada vez que se requiera un cambio en un usuario.	
Prioridad del CU:	Alta	

Identificación del requerimiento:	SEG04	
Nombre CU:	Asignar Rol.	
Objetivo:	Definir los niveles de acceso de los usuarios al sistema.	
Descripción.	Los usuarios activos del sistema, tienen capacidades funcionales según las tareas que van a desempeñar en el sistema.	
Precondición.	El usuario debe estar creado previamente. El administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema”	
Flujo de Éxito.	Paso	Acción
	1	El administrador del sistema ingresa al módulo de gestión de usuario y elige la opción crear nuevo usuario.
	2	Llena los datos (Nombres, Apellidos, Correo, Login, la Contraseña)
	3	El administrador define el Rol a desempeñar en el sistema
	4	El sistema verifica los nuevos permisos y los asigna al nuevo usuario.
Flujo Alternativo.	Paso	Acción
Pos-condición	El rol del usuario del sistema ha sido asignado	
Frecuencia.	Cada vez que un usuario de sistema es creado.	
Prioridad del CU:	Alta	

Identificación del requerimiento:	SEG05	
Nombre CU:	Gestión de Roles.	
Objetivo:	Definir los niveles de acceso según rol para el acceso al sistema.	
Descripción.	El Administrador del sistema podrá describir las capacidades que tendrán los roles para ser asignados a los usuarios del sistema.	
Precondición.	El administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema” Crear rol: No hay precondición. Actualiza: El rol debe existir.	
Flujo de Éxito.	Paso	Acción
	1	El administrador del sistema ingresa al módulo de gestión de roles y elige la opción crear o modificar un rol.
	2	Llena los datos del nuevo rol, los niveles de acceso, además definen el estado como activo o inactivo.
	3	El sistema verifica los datos y permisos del nuevo rol, y los almacena
Flujo Alternativo.	Paso	Acción
	3	Si los datos no son válidos, se solicita la corrección antes de almacenar el rol
Pos-condición	El rol está listo para ser asignado	
Puntos de Ampliación	“Crear Rol”, “Actualizar Rol”	
Frecuencia.	Cada que se quiera definir un nivel de acceso.	
Prioridad del CU:	Alta	

Identificación del requerimiento:	SEG06	
Nombre CU:	Ingresar al Sistema.	
Objetivo:	Autenticarse como usuario autorizado para utilizar el sistema	
Descripción.	Los usuarios activos del sistema, ingresan su nombre de usuario y su respectiva contraseña para realizar las operaciones asignadas según su rol en el sistema.	
Precondición.	Ser un usuario debidamente registrado en el sistema, además de tener un estado activo.	
Flujo de Éxito.	Paso	Acción
	1	El usuario abre la aplicación.
	2	El sistema solicita autenticación(Usuario y Contraseña)
	3	El usuario Ingresa sus credenciales.
	4	Se valida al usuario que este activo y que sus credenciales sean correctas, se concede el acceso al cual tiene permiso.
Flujo Alternativo.	Paso	Acción
	4	Se valida al usuario, el usuario no tiene el estado activo, o ingreso mal sus credenciales, se niega el acceso al sistema.
Pos-condición	El usuario ha sido autenticado correctamente y ahora puede utilizar el sistema.	
Frecuencia.	Cada vez que un usuario intente acceder al sistema.	
Prioridad del CU:	Alta	

Módulo de Ingresos y Gastos.

Identificación del requerimiento:	IG01	
Nombre CU:	Gestión de Ingresos y Gastos.	
Objetivo:	Administrar, organizar y definir la estructura necesaria para registrar y consultar los movimientos financieros.	
Descripción.	Permite registrar de forma organizada el uso de fondos con los que cuenta la FESASURF para realizar sus actividades.	
Precondición.	El administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema”	
Flujo de Éxito.	Paso	Acción
	1	El administrador del sistema ingresa al módulo de ingresos y gastos.
	2	El administrador tiene cuatro opciones: 1. Gestionar las cuentas. (USA Caso de Uso: Gestión de Cuentas IG02) 2. Gestionar los Patrocinadores (Extiende Caso de Uso: Gestión de Patrocinadores IG05) 3. Registrar las Transacciones (USA Caso de Uso: Registrar Transacción) 4. Consultar los saldos de las cuentas (Usa Caso de Uso: Consultar Saldos)
Flujo Alternativo.	Paso	Acción
Pos-condición	• Las cuentas han sido gestionadas: Creadas o actualizadas. • Los Patrocinadores han sido gestionados: Creados o actualizados. • Las transacciones han sido registrados.	
Puntos de Ampliación	“Gestión de Cuentas”, “Consultar Saldos”, “Gestión de Patrocinadores”, “Registrar Transacción”	
Frecuencia.	Cada vez que sea necesario registrar o visualizar una operación financiera	
Prioridad del CU:	Alta	

Identificación del requerimiento:	IG02	
Nombre CU:	Gestión de Cuentas.	
Objetivo:	Administrar la definición de las cuentas que serán utilizadas para detallar los movimientos financieros.	
Descripción.	Permite definir la estructura que será el contenedor para registrar los movimientos financieros de la FESASURF	
Precondición.	El administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema”	
Flujo de Éxito.	Paso	Acción
	1	El administrador del sistema ingresa al módulo de ingresos y gastos, elige la opción Gestión de Cuentas.
	2	El administrador tiene dos opciones: 1. Crear Cuenta. (USA Caso de Uso: Crear Cuenta IG03) 2. Actualizar Cuenta. (USA Caso de Uso: Actualizar Cuenta IG04)
Flujo Alternativo.	Paso	Acción
Pos-condición	Se crean las cuentas o son actualizadas las existentes.	
Puntos de Ampliación	“Crear Cuenta”, “Actualizar Cuenta”	
Frecuencia.	Cada vez que sea necesario crear una nueva cuenta.	
Prioridad del CU:	Alta	

Identificación del requerimiento:	IG03	
Nombre CU:	Crear Cuentas.	
Objetivo:	Definir las cuentas para registrar las transacciones.	
Descripción.	Crear la estructura que será el contenedor para registrar los movimientos financieros de la FESASURF	
Precondición.	El administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema”	
Flujo de Éxito.	Paso	Acción
	1	El administrador del sistema ingresa al módulo de ingresos y gastos, elige la opción crear nueva cuenta.
	2	El administrador define la naturaleza de la cuenta y sus atributos que la identifican
	3	El sistema valida los datos, y guarda la nueva cuenta
Flujo Alternativo.	Paso	Acción
	3	Si la cuenta ya existe, se solicita la corrección, se valida y se guarda al nueva cuenta.
Pos-condición	La cuenta ha sido creada y está lista para ser usada para registrar las transacciones.	
Frecuencia.	Cada vez que sea necesario crear una nueva cuenta.	
Prioridad del CU:	Alta	

Identificación del requerimiento:	IG04	
Nombre CU:	Actualizar Cuentas.	
Objetivo:	Modificar las cuentas en las que se requieran cambios en su estructura.	
Descripción.	Cambiar la estructura de una o varias cuentas para adecuarse a las necesidades de activar o desactivar una cuenta.	
Precondición.	El administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema” La cuenta debe existir previamente.	
Flujo de Éxito.	Paso	Acción
	1	El administrador del sistema ingresa al módulo de ingresos y gastos, elige la opción actualizar cuenta.
	2	El administrador puede cambiar la naturaleza de la cuenta y sus atributos que la identifican.
	3	El sistema valida los datos, y guarda la cuenta modificada
Flujo Alternativo.	Paso	Acción
	3	Si hay errores en los atributos, se solicita la corrección, se valida y se guarda la cuenta.
Pos-condición	La cuenta ha sido modificada y está lista para ser usada para registrar las transacciones.	
Frecuencia.	Cada vez que sea necesario modificar una cuenta.	
Prioridad del CU:	Alta	

Identificación del requerimiento:	IG05	
Nombre CU:	Gestionar Patrocinadores.	
Objetivo:	Almacenar la información de las entidades que apoyan a la FESASURF con recursos económicos.	
Descripción.	Definir los perfiles de las entidades o personas que donan recursos y almacenar un histórico de cada aportación.	
Precondición.	El administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema”	
Flujo de Éxito.	Paso	Acción
	1	El administrador del sistema ingresa al módulo de ingresos y gastos -> Gestionar Patrocinadores Tiene dos opciones: 1. Crear un patrocinador nuevo (Usa Caso de Uso: Crear Patrocinador IG06). 2. Actualizar un patrocinador existente (Usa Caso de Uso: Actualizar Patrocinador IG07).
Flujo Alternativo.	Paso	Acción
Pos-condición	El patrocinador ya tiene su perfil o se ha modificado, y ahora podrá ser usado para almacenar sus aportaciones	
Puntos de Ampliación	“Crear Patrocinador”, “Actualizar Patrocinador”	
Frecuencia.	Cada vez que sea necesario crear o actualizar un patrocinador.	
Prioridad del CU:	Alta	

Identificación del requerimiento:	IG06	
Nombre CU:	Crear Patrocinador.	
Objetivo:	Crear el perfil de las entidades que apoyan a la FESASURF y así poder almacenar su información.	
Descripción.	Se busca establecer el perfil de cada una de las instituciones o personas que aportan recursos económicos o materiales a la FESASURF.	
Precondición.	El administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema”	
Flujo de Éxito.	Paso	Acción
	1	El administrador del sistema ingresa al módulo de ingresos y gastos -> Gestionar Patrocinadores, elige la opción crear un patrocinador.
	2	Llenar los campos que identifican al patrocinador.
	3	El sistema valida los datos, y guarda el patrocinador
Flujo Alternativo.	Paso	Acción
	3	Si hay errores, se solicita la corrección, se valida y se guarda al patrocinador.
Pos-condición	El patrocinador ya tiene su perfil, y ahora podrá ser usado para almacenar sus aportaciones	
Frecuencia.	Cada vez que sea necesario crear un patrocinador.	
Prioridad del CU:	Alta	

Identificación del requerimiento:	IG07	
Nombre CU:	Actualizar Patrocinador.	
Objetivo:	Mantener actualizada la información de las entidades que apoyan a la FESASURF con recursos económicos.	
Descripción.	Nos permite hacer cambios a los atributos de las entidades o personas que donan recursos y almacenar un histórico de cada aportación, como cambios de contactos, dirección.	
Precondición.	El administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema” Deberá existir previamente el patrocinador.	
Flujo de Éxito.	Paso	Acción
	1	El administrador del sistema ingresa al módulo de ingresos y gastos -> Gestionar Patrocinadores, elige la opción actualizar patrocinador.
	2	Modificar los campos que necesitan ser actualizados
	3	El sistema valida los datos, y actualiza el registro del patrocinador
Flujo Alternativo.	Paso	Acción
	3	Si hay errores, se solicita la corrección, se valida y se actualiza al patrocinador.
Pos-condición	El perfil del patrocinador ha sido actualizado, y ahora podrá ser usado para almacenar sus aportaciones	
Frecuencia.	Cada vez que sea necesario actualizar un patrocinador.	
Prioridad del CU:	Alta	

Identificación del requerimiento:	IG08	
Nombre CU:	Registrar Transacción.	
Objetivo:	Guardar el detalle de cómo fueron invertidos los fondos de la FESASURF.	
Descripción.	Almacenar de forma detalla cómo fueron utilizados los recursos económicos con los que cuenta la FESASURF para realizar sus actividades.	
Precondición.	El administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema” Contar con la documentación debidamente autorizada por el contador	
Flujo de Éxito.	Paso	Acción
	1	El administrador del sistema ingresa al módulo de ingresos y gastos, elige la opción registrar transacción.
	2	El administrador del sistema asigna los ingresos o gastos de dinero conforme a los documentos y a la naturaleza de la cuenta que corresponda. 1. Ver Cuentas 2. Seleccionar Cuenta 3. Crear Transacción con los siguientes datos: Fecha, Comprobante, Forma de Pago, Referencia, Descripción, Monto, Tipo Transacción y Cuenta Seleccionada.
	3	El sistema valida los datos, y guarda la transacción.
Flujo Alternativo.	Paso	Acción
Pos-condición	La transacción ha sido debidamente registrada	
Frecuencia.	Cada vez que sea necesario crear una nueva cuenta.	
Prioridad del CU:	Alta	

Identificación del requerimiento:	IG09	
Nombre CU:	Registrar Ingresos.	
Objetivo:	Guardar el detalle de cómo se recibieron los fondos que serán invertidos en las actividades de la FESASURF.	
Descripción.	Almacenar de forma detalla las cantidades de dinero que ingresan como recursos económicos para la FESASURF.	
Precondición.	El Administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema” Contar con la documentación debidamente autorizada por el contador	
Flujo de Éxito.	Paso	Acción
	1	El administrador del sistema ingresa al módulo de ingresos y gastos, elige la opción registrar transacción -> Registrar Ingresos.
	2	El usuario asigna los cargos de dinero conforme a los documentos y a la naturaleza de la cuenta que corresponda. Se tienen dos opciones 1. Registrar Aportaciones (USA Caso de Uso: Aportaciones IG10). 2. Registrar Presupuesto INDES (USA Caso de Uso: Presupuesto INDES IG11).
	3	El sistema valida los datos, y guarda la transacción.
Flujo Alternativo.	Paso	Acción
Pos-condición	La transacción ha sido debidamente registrada	
Puntos de Ampliación	“Aportaciones”, “Presupuesto INDES”	
Frecuencia.	Cada vez que sea necesario registrar un ingreso de recursos económicos.	
Prioridad del CU:	Alta	

Identificación del requerimiento:	IG10	
Nombre CU:	Aportaciones.	
Objetivo:	Guardar el detalle de los fondos recibidos por las instituciones o personas que apoyan con recursos las actividades de la FESASURF.	
Descripción.	Almacenar de forma detalla las cantidades de dinero que ingresan como recursos económicos para la FESASURF.	
Precondición.	El Administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema” Contar con la documentación debidamente autorizada por el contador	
Flujo de Éxito.	Paso	Acción
	1	El Administrador del sistema ingresa al módulo de ingresos y gastos, elige la opción registrar transacción -> Registrar Ingresos -> Aportaciones.
	2	El Administrador del sistema asigna los cargos de dinero conforme a los documentos y a la naturaleza de la cuenta que corresponda.
	3	El sistema valida los datos, y guarda la aportación.
Flujo Alternativo.	Paso	Acción
Pos-condición	La aportación ha sido debidamente registrada	
Frecuencia.	Cada vez que sea necesario registrar una aportación de recursos económicos.	
Prioridad del CU:	Alta	

Identificación del requerimiento:	IG11	
Nombre CU:	Presupuesto INDES.	
Objetivo:	Guardar el detalle de los fondos recibidos del presupuesto que asigna INDES que serán invertidos en las actividades de la FESASURF.	
Descripción.	Almacenar de forma detalla el presupuesto INDES como recursos económicos para la FESASURF.	
Precondición.	El Administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema” Contar con la documentación debidamente autorizada por el contador	
Flujo de Éxito.	Paso	Acción
	1	El Administrador del sistema ingresa al módulo de ingresos y gastos, elige la opción registrar transacción -> Registrar Ingresos- >Presupuesto INDES.
	2	El Administrador del sistema asigna los cargos de dinero conforme a los documentos y a la naturaleza de la cuenta que corresponda los fondos INDES.
	3	El sistema valida los datos, y guarda la transacción.
Flujo Alternativo.	Paso	Acción
Pos-condición	El presupuesto INDES ha sido debidamente registrado	
Frecuencia.	Cada vez que sea necesario registrar el presupuesto INDES	
Prioridad del CU:	Alta	

Identificación del requerimiento:	IG12	
Nombre CU:	Registrar Gastos.	
Objetivo:	Guardar el detalle de cómo se invierten los fondos que son utilizados en las actividades de la FESASURF.	
Descripción.	Almacenar de forma detalla las cantidades de dinero que salen como gastos de recursos económicos para la FESASURF durante la realización de sus actividades.	
Precondición.	El Administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema” Contar con la documentación debidamente autorizada por el contador	
Flujo de Éxito.	Paso	Acción
	1	El Administrador del sistema ingresa al módulo de ingresos y gastos, elige la opción registrar transacción -> Registrar Gastos.
	2	El Administrador del sistema asigna las salidas de dinero conforme a los documentos y a la naturaleza de la cuenta que corresponda.
	3	El sistema valida los datos, y guarda la transacción.
Flujo Alternativo.	Paso	Acción
Pos-condición	Los gastos ha sido debidamente registrados	
Frecuencia.	Cada vez que sea necesario registrar un gasto de recursos económicos.	
Prioridad del CU:	Alta	

Identificación del requerimiento:	IG13	
Nombre CU:	Consultar Saldos.	
Objetivo:	Mostrar el movimiento que han tenido las cuentas de ingresos o de gastos.	
Descripción.	Permitirá visualizar las transacciones realizas durante el periodo de operaciones	
Precondición.	El Administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema”	
Flujo de Éxito.	Paso	Acción
	1	El Administrador del sistema ingresa al módulo de Ingresos y Gastos -> Gestionar Patrocinadores, elige la opción de Consultar Saldos.
	2	El Administrador del sistema puede elegir visualizar las cuentas Se tienen dos opciones: 1. Cuentas Ingresos (USA Caso de Uso: Cuentas Ingresos IG14). 2. Cuentas Gastos (USA Caso de Uso: Cuentas Gastos IG15).
	3	El sistema presenta los datos solicitados
Flujo Alternativo.	Paso	Acción
Pos-condición	El usuario ya cuenta con la información financiera de las operaciones realizadas	
Puntos de Ampliación	“Cuentas Ingresos”, “Cuentas Gastos”	
Frecuencia.	Cada vez que sea necesario consultar los movimientos de las cuentas.	
Prioridad del CU:	Alta	

Identificación del requerimiento:	IG14	
Nombre CU:	Cuentas Ingresos.	
Objetivo:	Mostrar el movimiento que han tenido las cuentas de ingresos.	
Descripción.	Permitirá visualizar las transacciones realizas durante el periodo de operaciones	
Precondición.	El Administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema”	
Flujo de Éxito.	Paso	Acción
	1	El Administrador del sistema ingresa al módulo de Ingresos y Gastos -> Gestionar Patrocinadores, elige la opción de Consultar Saldos -> Cuentas Ingresos.
	2	El Administrador del sistema puede visualizar las cuentas de Ingresos individualmente.
	3	El sistema presenta los datos solicitados
Flujo Alternativo.	Paso	Acción
Pos-condición	El usuario ya cuenta con la información financiera de las operaciones realizadas	
Frecuencia.	Cada vez que sea necesario consultar los movimientos de las cuentas de ingresos.	
Prioridad del CU:	Alta	

Identificación del requerimiento:	IG15	
Nombre CU:	Cuentas Gastos.	
Objetivo:	Mostrar el movimiento que han tenido las cuentas de Gastos.	
Descripción.	Permitirá visualizar las transacciones realizas durante el periodo de operaciones	
Precondición.	El Administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema”	
Flujo de Éxito.	Paso	Acción
	1	El Administrador del sistema ingresa al módulo de Ingresos y Gastos -> Gestionar Patrocinadores, elige la opción de Consultar Saldos -> Cuentas Gastos.
	2	El Administrador del sistema puede visualizar las cuentas de Gastos individualmente.
	3	El sistema presenta los datos solicitados
Flujo Alternativo.	Paso	Acción
Pos-condición	El usuario ya cuenta con la información financiera de las operaciones realizadas	
Frecuencia.	Cada vez que sea necesario consultar los movimientos de las cuentas de gastos.	
Prioridad del CU:	Alta	

Módulo de Competencias.

Identificación del requerimiento:	COM01
Nombre CU:	Agregar Categorías
Objetivo:	Adicionar nuevas categorías de atletas para competencias en la FESASURF
Descripción.	El Administrador del sistema podrá agregar nuevas categorías como los son (Júnior, Longboard, Damas, Bodyboard, Master, Open, Groms)

Precondición.	El Administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema”	
Flujo de Éxito.	Paso	Acción
	1	El administrador del sistema ingresa al módulo de Competencias y selecciona la opción Gestión de Categorías.
	2	Selecciona Agregar Categoría
	3	Llena los datos generales de la categoría: el nombre de la categoría, edad mínima y máxima para estar en la categoría, sexo, una descripción y selecciona Crear
	4	El sistema realiza una validación de los datos de la nueva categoría y los almacena
Flujo Alternativo.	Paso	Acción
	4	Si los datos tienen inconsistencias, se solicita la corrección, se valida y los almacena
Pos-condición	La categoría ha sido creada y está lista para asignarse en las competencias	
Puntos de Ampliación	“Crear Circuito”	
Frecuencia.	Cada vez que se necesite crear una categoría	
Prioridad del CU:	Media	

Identificación del requerimiento:	COM02	
Nombre CU:	Modificar Categorías	
Objetivo:	Realizar los cambios en los datos de las categorías para competencias en la FESASURF	
Descripción.	El Administrador del sistema podrá modificar las categorías de atletas	
Precondición.	El Administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema”	
Flujo de Éxito.	Paso	Acción
	1	El administrador del sistema ingresa al módulo de Competencias y selecciona la opción Gestión de Categorías.
	2	Selecciona Modificar Categoría
	3	Modifica los datos generales de la categoría: el nombre de la categoría, edad mínima y máxima para estar en la categoría, sexo, una descripción y selecciona modificar
	4	El sistema realiza una validación de los datos de la categoría y los modifica
Flujo Alternativo.	Paso	Acción
	4	Si los datos tienen inconsistencias, se solicita la corrección, se valida y los almacena
Pos-condición	La categoría ha sido modificada y está lista para asignarse en las competencias	
Frecuencia.	Cada vez que se necesite modificar una categoría	
Prioridad del CU:	Media	

Identificación del requerimiento:	COM03	
Nombre CU:	Crear Circuitos	
Objetivo:	Agregar nuevos circuitos (competencias) para rankear los mejores atletas	
Descripción.	El Administrador del sistema podrá agregar nuevos circuitos teniendo en cuenta que los circuitos incluyen tanto rondas, heats, puntuaciones así como también atletas y jueces participantes en cada una de ellos que deben ser agregados posterior a la creación de este.	
Precondición.	El Administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema”	
Flujo de Éxito.	Paso	Acción
	1	El administrador del sistema ingresa al módulo de Competencias, selecciona la opción Gestión de Circuitos y elige Crear Circuito
	2	Especifica un nombre, lugar, fecha, categoría y una descripción del circuito.
	3	Selecciona Crear
	4	El sistema realiza una validación de los datos del circuito y los almacena
Flujo Alternativo.	Paso	Acción
	2	Especifica los datos del Circuito y Elige Gestión de Categorías en caso de agregar una categoría no existente
	3	Selecciona Crear
	4	El sistema realiza una validación de los datos del circuito y los almacena
Flujo Alternativo.	Paso	Acción
	4	Si los datos tienen inconsistencias, se solicita la corrección, se valida y los almacena
Pos-condición	El circuito ha sido creado y consolidado con el ranking	
Puntos de Ampliación	“Gestionar Categoría”	
Frecuencia.	Cada vez que se necesite crear un circuito	
Prioridad del CU:	Alta	

Identificación del requerimiento:	COM04	
Nombre CU:	Modificar Circuitos	
Objetivo:	Editar Circuitos (competencias) existentes	
Descripción.	El Administrador del sistema podrá modificar circuitos teniendo en cuenta que los circuitos modificados incluyen tanto rondas, heats, puntuaciones así como también atletas y jueces participantes en cada una de ellos que pueden ser afectados	
Precondición.	El Administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema” El circuito a modificar ya debe estar creado anteriormente	
Flujo de Éxito.	Paso	Acción
	1	El administrador del sistema ingresa al módulo de Competencias y selecciona la opción Gestión de Circuitos.
	2	Selecciona Modificar Circuito y elige el circuito a editar
	3	Modifica los datos del Circuito (nombre, lugar, fecha, categoría y descripción del circuito)
	4	Selecciona Modificar
	5	El sistema realiza una validación de los datos del circuito y los Modifica.
Flujo Alternativo.	Paso	Acción
	3	Modifica los datos del Circuito y Elige Gestión de Categorías en caso de agregar una categoría no existente
	4	Selecciona Modificar
	5	El sistema realiza una validación de los datos del circuito y los Modifica.
Flujo Alternativo.	Paso	Acción
	5	Si los datos tienen inconsistencias, se solicita la corrección, se valida y los modifica
Pos-condición	El circuito ha sido editado y consolidado con el ranking	
Puntos de Ampliación	“Gestionar Categoría”	
Frecuencia.	Cada vez que se necesite editar un circuito	
Prioridad del CU:	Alta	

Identificación del requerimiento:	COM05	
Nombre CU:	Consultar Ranking	
Objetivo:	Consultar el ranking global de los atletas	
Descripción.	El Administrador del sistema podrá consultar el ranking global por categorías (Júnior, Longboard, Damas, Bodyboard, Master, Open, Groms) de las competencias realizas en la FESASURF	
Precondición.	El Administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema”	
Flujo de Éxito.	Paso	Acción
	1	El administrador del sistema ingresa al módulo de Competencias
	2	Selecciona la opción Consultar Ranking
	3	El sistema lista en ranking global de los atletas permitiendo filtrarlo por competencia y categoría
	4	El sistema despliega la opción de imprimir como reporte dicho ranking
Flujo Alternativo.	Paso	Acción
Pos-condición	El ranking global ha sido desplegado y está listo para imprimirse	
Frecuencia.	Cada vez que se necesite consultar el ranking	
Prioridad del CU:	Alta	

Identificación del requerimiento:	COM06	
Nombre CU:	Agregar Jueces participantes	
Objetivo:	Asignar jueces evaluadores para cada circuito	
Descripción.	El Administrador del sistema podrá asignar jueces a cada circuito, con sus respectivos roles	
Precondición.	El Administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema” El juez ya debe de estar registrado dentro del sistema. El circuito ya debe de estar creado.	
Flujo de Éxito.	Paso	Acción
	1	El administrador de sistema ingresa al módulo de competencias y en la gestión de circuitos
	2	Selecciona un circuito ya creado, elige gestión de jueces participantes y luego usa la opción agregar jueces participantes
	3	Selecciona los jueces que van a participar en él.
	4	Después de haber seleccionado los jueces será necesario establecer el rol que desarrollaran dentro del circuito. Estos pueden ser: • Beach Marshall • Juez Mayor • Jueces evaluadores • Spotter
	5	El sistema realiza una validación de los datos ingresados y se procede a guardar los datos
Flujo Alternativo.		
Pos-condición	Los jueces ya podrán acceder al sistema y ver los datos del circuito.	
Frecuencia.	Cada vez que se realice una circuito.	
Prioridad del CU:	Alta	

Identificación del requerimiento:	COM07	
Nombre CU:	Gestión de Heats	
Objetivo:	Generar y definir los Heats para cada ronda	
Descripción.	El Administrador del sistema podrá definir Heats de forma dinámica, dependiendo la cantidad de atletas registrados para cada categoría, tomando en consideración que cada Heat debe de tener como mínimo tres atletas y máximo 4, aunque existe el caso excepcional de tener un máximo de 5 atletas en un Heat.	
Precondición.	El Administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema” La ronda ya debe estar creada Los atletas deben de estar previamente inscritos.	
Flujo de Éxito.	Paso	Acción
	1	El administrador del sistema ingresa al módulo de competencias y en la gestión de circuitos, selecciona el circuito.
	2	Selecciona la opción Gestión de Rondas
	3	Selecciona Gestión de Heats
	4	El administrador tiene dos opciones: 1 Generar Heat (Usa Caso de Uso: Generar Heat.) 2.Modificar Heat(USA Caso de Uso: Modificar Heat)
Flujo Alternativo.		
Pos-condición	Cada ronda ya tendrá los Heats asignados	
Puntos de Ampliación	“Gestión de Puntuaciones”, “Asignar Jueces”, “Asignar Atletas”	
Frecuencia.	Cada vez que se realice una ronda dentro de una circuito.	
Prioridad del CU:	Alta	

Identificación del requerimiento:	COM08	
Nombre CU:	Generar Heats	
Objetivo:	Generar Heats para cada ronda	
Descripción.	El Administrador del sistema podrá crear Heats de forma dinámica, dependiendo la cantidad de atletas registrados para cada categoría, tomando en consideración que cada Heat debe de tener como mínimo tres atletas y máximo 4, aunque existe el caso excepcional de tener un máximo de 5 atletas en un Heat.	
Precondición.	El Administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema” La ronda ya debe estar creada Los atletas deben de estar previamente inscritos.	
Flujo de Éxito.	Paso	Acción
	1	El administrador del sistema ingresa al módulo de competencias y en la gestión de circuitos, selecciona el circuito.
	2	Selecciona la opción Gestión de Rondas y la ronda requerida
	3	Selecciona la gestión de Heats y luego generar Heats
	4	El sistema buscara automáticamente quienes serán las cabezas de heat, es decir los atletas que tengan mejor rango no podrán competir entre ellos en las primeras rondas
	5	Teniendo las cabezas de Heat el sistema procederá a distribuir uniformemente los atletas por cada Heat y asignándoles un color para poder diferenciarlos.
	6	El sistema muestra los Heats generados con los atletas asignados para cada uno de ellos
Flujo Alternativo.		
Pos-condición	La ronda ya tendrá los Heats asignados	
Puntos de Ampliación	“Gestión de Puntuaciones”, “Asignar Jueces”, “Asignar Atletas”	
Frecuencia.	Cada vez que se realice una ronda dentro de un circuito.	
Prioridad del CU:	Alta	

Identificación del requerimiento:	COM09	
Nombre CU:	Modificar Heat	
Objetivo:	Realizar modificaciones sobre los Heats creados.	
Descripción.	El Administrador del sistema podrá modificar la información sobre los Heats almacenados	
Precondición.	El Administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema” El Heat debe estar previamente creado	
Flujo de Éxito.	Paso	Acción
	1	El administrador del sistema ingresa al módulo de competencias, selecciona el circuito y ronda
	2	Elige el Heat y selecciona la opción Modificar
	3	Modifica los datos del Heat y confirma la operación al sistema
	4	El sistema realiza una validación de los datos del Heat seleccionado y realizara la actualización de datos
Pos-condición	El Heat se ha modificado exitosamente	
Frecuencia.	Cada vez que sea necesario modificar un Heat	
Prioridad del CU:	Alta	

Identificación del requerimiento:	COM10	
Nombre CU:	Asignar jueces (por Heat)	
Objetivo:	Asignar que jueces evaluaran cada uno de los Heats	
Descripción.	El Administrador del sistema podrá asignar jueces a cada Heat, actualmente son 4 jueces evaluadores de los cuales solo tres evalúan el Heat, en el siguiente Heat entra el juez que estaba descansando y sale uno de lo que estaban evaluando, este proceso debe de realizarse para todos los Heats.	
Precondición.	El Administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema” El juez ya debe de estar registrado dentro del sistema. El circuito ya debe de estar creado. La ronda ya debe estar creada	
Flujo de Éxito.	Paso	Acción
	1	El administrador del sistema ingresa a la gestión de circuitos, elige el circuito y continua en gestión de rondas
	2	Selecciona una ronda
	3	Selecciona generar Heats (caso de uso generar Heats)
	4	Una vez generados los Heats el sistema automáticamente procederá a realizar la asignación de jueces evaluadores y sus roles. Para cada Heat creado deberá de asignarse tres jueces evaluadores de 4 disponibles, el 4 cuarto juez descansara un turno, y en el siguiente Heat sustituirá a un juez que está evaluando, este ciclo se repetirá para cada Heat.
	5	El sistema realiza una validación de los datos en la asignación de jueces por cada Heat y se procederá a guardar los datos
Flujo Alter.		
Pos-condición	Los jueces ya podrán saber que Heat evaluaran.	
Puntos de Ampliación	“Asignar Rol”	
Frecuencia.	Cada vez que se cree un Heat	
Prioridad del CU:	Alta	

Identificación del requerimiento:	COM11	
Nombre CU:	Asignar Atletas (por Heat)	
Objetivo:	Asignar que los atletas que van a participar en cada Heat.	
Descripción.	Dependiendo del número de competidores en una Ronda se distribuyen en Heats, de la siguiente forma: mínimo por Heat 3 competidores, pasan los 2 mejores por Heat, por ejemplo: si hay 9 competidores se pueden distribuir 3 Heats de tres integrantes cada uno, si solo hay 5 competidores se integran en un solo Heat, definitivo y se califican los mejores puntajes	
Precondición.	El Administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema” El circuito ya debe de estar creado. La ronda ya debe estar creada	
Flujo de Éxito.	Paso	Acción
	1	El administrador del sistema ingresa a la gestión de circuitos, elige el circuito y continua en gestión de rondas
	2	Selecciona una ronda
	3	Selecciona generar Heats (caso de uso generar Heats)
	4	Una vez generados los Heats el sistema automáticamente distribuirá los atletas en los Heats. Para cada Heat creado deberán asignar mínimo 3 atletas y un máximo de 4, en caso de pocos atletas 5 en total, se asignaran todos al mismo Heat.
	5	El sistema realiza una validación de los datos en la asignación de atletas por cada Heat y se procederá a guardar los datos
Flujo Alternativo.		
Pos-condición	Los Heats ya tienen los atletas que van a participar en cada uno	
Frecuencia.	Cada vez que se deba asignar atletas a un Heat	
Prioridad del CU:	Alta	

Identificación del requerimiento:	COM12	
Nombre CU:	Gestión de Puntuaciones	
Objetivo:	El sistema debe poder (Asignar, Modificar) Puntaciones asociadas a las rondas y a los Heats	
Descripción.	Se deben poder guardar puntaciones de todos los surfistas que participen en competencia y se debe poder visualizar y filtrar estas puntuaciones, los filtros pueden ser por competencia(circuito), por ronda, por Heat, por Surfista, por categoría, por fecha y deben poder ordenarse por el valor de la puntuación	
Precondición.	El juez debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema” El juez ya debe de estar registrado dentro del sistema. El circuito ya debe de estar creado. La ronda ya debe estar creada Él Heat ya debe estar generado	
Flujo de Éxito.	Paso	Acción
	1	El juez ingresa a la Gestión de Puntuaciones de un circuito en especifico
	2	Selecciona una ronda y un Heat
	3	Se muestra el Heat a calificar en forma de matriz de puntuaciones que contiene (color del atleta participante y las olas numeradas del 1 al 10)
	3	El juez selecciona una casilla de nota y tiene dos opciones: 1. Agregar Puntuación (USA Caso de Uso: Agregar Puntuación). 2. Modificar Puntuación (USA Caso de Uso: Modificar Puntuación.)
Flujo Alternativo.		
Pos-condición	El juez asigno sus puntuaciones por Heat	
Frecuencia.	Cada vez que se deba asignar puntuaciones	
Prioridad del CU:	Alta	

Identificación del requerimiento:	COM13	
Nombre CU:	Agregar Puntuaciones	
Objetivo:	El sistema debe poder asignar las puntaciones asociadas a las rondas y a los Heats de un surfista	
Descripción.	Se deben poder asignar puntuaciones por cada Heat a todos los surfistas que participen en competencia	
Precondición.	El juez ya debe de estar registrado dentro del sistema. El juez debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema” El circuito ya debe de estar creado. La ronda ya debe estar creada El Heat ya debe estar generado	
Flujo de Éxito.	Paso	Acción
	1	El juez ingresa a la Gestión de Puntuaciones de un circuito en especifico
	2	Selecciona una ronda y un Heat
	3	Seleccionando cada casilla de notas y agregándolas, El juez se encarga de llenar la matriz de puntuaciones que los atleta con colores asignados consigan en su participación en cada Heat, estas posteriormente será utilizada para ser procesada y descartar los puntajes que no se toman en cuenta. (Extiende Caso de Uso Calculo de Puntuaciones)
Flujo Alternativo.		
Pos-condición	Los atletas tienen asignadas sus puntuaciones por Heats.	
Frecuencia.	Cada vez que se deba asignar puntuaciones	
Prioridad del CU:	Alta	

Identificación del requerimiento:	COM14	
Nombre CU:	Modificar Puntuaciones	
Objetivo:	El sistema debe poder modificar las puntaciones asociadas a las rondas y a los Heats de un surfista	
Descripción.	Se deben poder modificar puntaciones de todos los surfistas que participen en competencia por cada Heat.	
Precondición.	El juez ya debe de estar registrado dentro del sistema. El juez debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema” El circuito ya debe de estar creado. La ronda ya debe estar creada El Heat ya debe estar generado. Las puntuaciones pueden ser modificadas en un periodo no mayor a 15 minutos después de haber sido publicadas	
Flujo de Éxito.	Paso	Acción
	1	El juez ingresa a la Gestión de Puntuaciones de un circuito en específico
	2	Selecciona una ronda y un Heat
	3	El juez se encarga de modificar la matriz de puntuaciones que posteriormente será utilizada para ser procesada y descartar los puntajes que no se toman en cuenta. (Extiende Caso de Uso Calculo de Puntuaciones)
Flujo Alternativo.		
Pos-condición	Los atletas tienen asignados sus puntuaciones por Heats.	
Frecuencia.	Cada vez que se deba modificar puntuaciones	
Prioridad del CU:	Alta	

Identificación del requerimiento:	COM15	
Nombre CU:	Calcular Puntuaciones	
Objetivo:	El sistema debe poder calcular las puntaciones asociadas a las rondas y a los Heats de un surfista	
Descripción.	Se deben poder calcular puntaciones de todos los surfistas que participen en competencia por cada Heat.	
Precondición.	El juez ya debe de estar registrado dentro del sistema. El juez debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema” El circuito ya debe de estar creado. La ronda ya debe estar creada El Heat ya debe estar generado El juez debe de haber asignado las puntuaciones y enviadas para ser procesadas.	
Flujo de Éxito.	Paso	Acción
	1	El juez ingresa a la opción Agregar Puntuaciones de un circuito en específico
	2	Selecciona una ronda y un Heat
	3	El juez se encarga de llenar la matriz de puntuaciones y envía para ser procesada.
	4.	Se promedia los puntajes obtenidos y se descartan el más alto y el más bajo, y se define el puntaje obtenido en el Heat por el atleta.
Flujo Alternativo.		
Pos-condición	Los atletas tienen calculados sus puntuaciones finales por Heats.	
Frecuencia.	Cada vez que se deba calcular puntuaciones	
Prioridad del CU:	Alta	

Identificación del requerimiento:	COM16	
Nombre CU:	Asignaciones de Puntos para el Ranking	
Objetivo:	El sistema debe asignar los puntos al ranking	
Descripción.	Siempre que se realice el cálculo de puntajes en un Heat exitosamente se debe actualizar automáticamente los puntos de la ronda a la que pertenece el Heat y de igual manera el ranking hasta el momento para poder apreciar las posiciones de los atletas participantes	
Precondición.	Los Heats deben haber finalizado	
Flujo de Éxito.	Paso	Acción
	1	Una vez calculadas las puntuaciones de cada Heat, los resultados son agregados a los perfiles de cada atleta
	2	El ranking por competencia se actualiza para listar las posiciones obtenidas por los participantes.
	3	El ranking general se actualiza para agregar los puntos obtenidos por atleta.
Flujo Alternativo.		
Pos-condición	Los atletas tienen actualizados sus puntuaciones en el ranking.	
Frecuencia.	Cada vez que se deba actualizar el ranking	
Prioridad del CU:	Alta	

Identificación del requerimiento:	COM17	
Nombre CU:	Inscribir atletas participantes	
Objetivo:	Agregar a los atletas participaran en una competencia	
Descripción.	Se deberá mostrar una vista en formato de tabla paginada con un buscador de atletas que permita agregar atletas a la misma si el atleta no existe debe poder agregarse (botón de referencia a agregar atleta)	
Precondición.	El Administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema” El Circuito ya debe estar creado	
Flujo de Éxito.	Paso	Acción
	1	El Administrador del sistema selecciona el circuito requerido dentro de la gestión de circuitos en el Modulo de competencia
	2	El sistema muestra la opciones del circuito
	3	El Administrador del sistema selecciona la opción gestionar atletas participantes y luego inscribir atletas participantes.
	4	El sistema muestra una pantalla con la lista de atletas y un buscador que permitirá seleccionar surfistas y agregarlos a competencias.
	5	El Administrador del sistema selecciona un atleta y presiona la opción agregar atletas
	6	El sistema agrega el atleta a la competencia
Flujo Alternativo.		
Pos-condición	El atleta está inscrito correctamente y listo para participar	
Frecuencia.	Cada vez que sea necesario Agregar un atleta a una competencia	
Prioridad del CU:	Alta	

Identificación del requerimiento:	COM18	
Nombre CU:	Desinscribir atletas participantes	
Objetivo:	Permite eliminar un surfista de una competencia desinscribiendolo de la misma	
Descripción.	Se deberá mostrar una vista en formato de tabla paginada con un buscador de atletas del circuito y que permita seleccionar un atleta para desinscribirlo	
Precondición.	El Administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema” El circuito debe estar creado El atleta debe estar inscrito al circuito	
Flujo de Éxito.	Paso	Acción
	1	El Administrador del sistema selecciona el circuito requerido dentro de la gestión de circuitos en el Módulo de competencia
	2	El sistema muestra la opciones del circuito
	3	El Administrador del sistema selecciona la opción gestionar atletas participantes y luego desinscribir atletas participantes.
	4	El sistema muestra una pantalla con la lista de atletas
	5	El Administrador del sistema selecciona un atleta y presiona la opción de desinscribir atletas
	6	El sistema elimina el atleta del circuito
Flujo Alternativo.		
Pos-condición	Eliminación de la asociación entre atleta y competencia ,actualización de pantalla	
Frecuencia.	Cada vez que sea necesario Desinscribir un atleta de una competencia	
Prioridad del CU:	Alta	

Identificación del requerimiento:	COM19	
Nombre CU:	Gestión de rondas	
Objetivo:	Gestionar las rondas que tendrá una competencia	
Descripción.	El Administrador del sistema podrá gestionar las rondas que contiene cada competencia	
Precondición.	El Administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema” El circuito ya debe estar creado .	
Flujo de Éxito.	Paso	Acción
	1	El administrador del sistema ingresa al módulo de competencias y en la gestión de circuitos, selecciona el circuito.
	2	Selecciona la opción Gestión de Rondas
	3	El administrador tiene dos opciones: 1. Crear Ronda (Usa Caso de Uso: Crear Ronda.) 2.Modificar Ronda(Usa Caso de Uso: Modificar Ronda)
Flujo Alternativo.		
Pos-condición		
Puntos de Ampliación	“Crear Rondas”, ”Modificar Rondas”, “Gestión de Atletas Participantes”, “Gestión de Jueces Participantes”	
Frecuencia.	Cada vez que sea inicie una competencia	
Prioridad del CU:	Alta	

Identificación del requerimiento:	COM20	
Nombre CU:	Crear rondas	
Objetivo:	Crear las rondas que tendrá una competencia	
Descripción.	El Administrador del sistema podrá crear las rondas que contiene cada competencia se deberán crear rondas cada vez que se termine una ronda y queden más de 6 participantes en la competencias	
Precondición.	El Administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema” El circuito ya debe estar creado.	
Flujo de Éxito.	Paso	Acción
	1	El Administrador del sistema Inicia el circuito
	2	El Administrador del sistema crea la ronda asignando la cantidad de surfistas que competirán en la ronda
	3	Se generan los Heat por cada ronda, se crearan tantos Heat como sean necesarios dependiendo el número de atletas
	4	Al finalizar todos los Heat el sistema calculara cuantos participantes han clasificado
	5	Se repetirán los pasos 2, 3 y 4 hasta que el número de participantes sea 4 y está seria la ronda final
Flujo Alternativo.		
Pos-condición		
Puntos de Ampliación	“Gestión de Heats”	
Frecuencia.	Cada vez que sea inicie una competencia o terminen los heat de una ronda	
Prioridad del CU:	Alta	

Identificación del requerimiento:	COM21	
Nombre CU:	Modificar rondas	
Objetivo:	Modificar las rondas que tendrá una competencia	
Descripción.	El Administrador del sistema podrá modificar las rondas que contiene cada competencia de ser necesario	
Precondición.	El Administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema” El circuito ya debe estar creado La ronda ya debe estar creada.	
Flujo de Éxito.	Paso	Acción
	1	El Administrador del sistema selecciona el circuito y la ronda a modificar
	2	El Administrador del sistema reasigna la cantidad de surfistas que competirán en la ronda
	3	Genera nuevamente los Heat que pertenecen a ella, y se crearan tantos Heat como sean necesarios dependiendo el número de atletas que se reasignaron
Flujo Alternativo.		
Pos-condición	La ronda ha sido modificada exitosamente	
Frecuencia.	Cada vez que sea necesario modificar una ronda	
Prioridad del CU:	Alta	

Módulo de Noticias

Identificación del requerimiento:	NT01	
Nombre CU:	Agregar Noticias	
Objetivo:	Crear noticias de interés de la FESASURF	
Descripción.	El Administrador del sistema podrá crear noticias para poder ser consumidas mediante la aplicación móvil o mediante el sitio web.	
Precondición.	El Administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema”	
Flujo de Éxito.	Paso	Acción
	1	El administrador de sistema ingresa al módulo de noticias
	2	Selecciona la opción crear noticia.
	3	Llena el formulario con la información pertinente de la noticia(Deberá de permitir subir imágenes y agregar contenido dinámico)
	4	El sistema realiza una validación de la información y se procede a almacenar la información
Flujo Alternativo.		
Pos-condición	Se ha creado la noticia	
Frecuencia.	Cada vez que se desee crear contenido	
Prioridad del CU:	Media	

Identificación del requerimiento:	NT02	
Nombre CU:	Modificar noticias	
Objetivo:	Realizar modificaciones sobre las noticias creadas.	
Descripción.	El Administrador del sistema podrá modificar la información sobre las noticias almacenadas	
Precondición.	El Administrador del sistema debe haber ingresado previamente al sistema, caso de uso “Ingresar al Sistema” La noticia debe estar previamente creada	
Flujo de Éxito.	Paso	Acción
	1	El administrador del sistema ingresa al módulo noticias
	2	Elige la noticia y selecciona la opción Modificar
	3	Modifica los datos de la noticia y confirma la operación al sistema
	4	El sistema realiza una validación de los datos de la noticia seleccionada y realizara la actualización de datos
Pos-condición	La noticia se ha modificado exitosamente.	
Frecuencia.	Cada vez que sea necesario modificar una noticia	
Prioridad del CU:	Media	

Identificación del requerimiento:	NT03	
Nombre CU:	Ver Noticias	
Objetivo:	Ver las noticias creadas.	
Descripción.	los usuarios podrán ver noticias creadas por la administración de la FESASURF	
Precondición.	La noticia debe estar previamente creada	
Flujo de Éxito.	Paso	Acción
	1	El usuario ingresa a la aplicación web
	2	En la pantalla principal se muestran las noticias creadas por orden de creación
	3	El usuario da click en una noticia para poder ver toda la información acerca de la misma
Flujo Alternativo.	1	El usuario ingresa a la aplicación móvil
	2	En la pantalla principal se muestran las noticias creadas por orden de creación
	3	El usuario selecciona una noticia para poder ver toda la información acerca de la misma
Pos-condición	La noticia se muestra correctamente	
Frecuencia.	Cada vez que un usuario desee ver una noticia	
Prioridad del CU:	Media	

Aplicación Móvil.

Identificación del requerimiento:	MOV01
Nombre CU:	Notificar usuario.
Objetivo:	Establecer un sistema de notificaciones a nivel de usuario.
Descripción.	Se debe generar una funcionalidad que permita que cuando falten n días para el inicio de una competencia se notifiquen a todos los usuario de la misma
Precondición.	El usuario de la aplicación móvil debe tener activas las notificaciones

Flujo de Éxito.	Paso	Acción
	1	El administrador de sistema agrega una nueva competencia.
	2	Un proceso periódico consulta todas las competencias y evalúa cuantos días faltan para su inicio.
	3	Si la competencia cumple con los n días configurables se envía una notificación al usuario.
Flujo Alternativo.		
Pos-condición	Los usuarios de la aplicación deben poder ver la competencia	
Frecuencia.	Cada vez que se realice una competencia.	
Prioridad del CU:	Media	

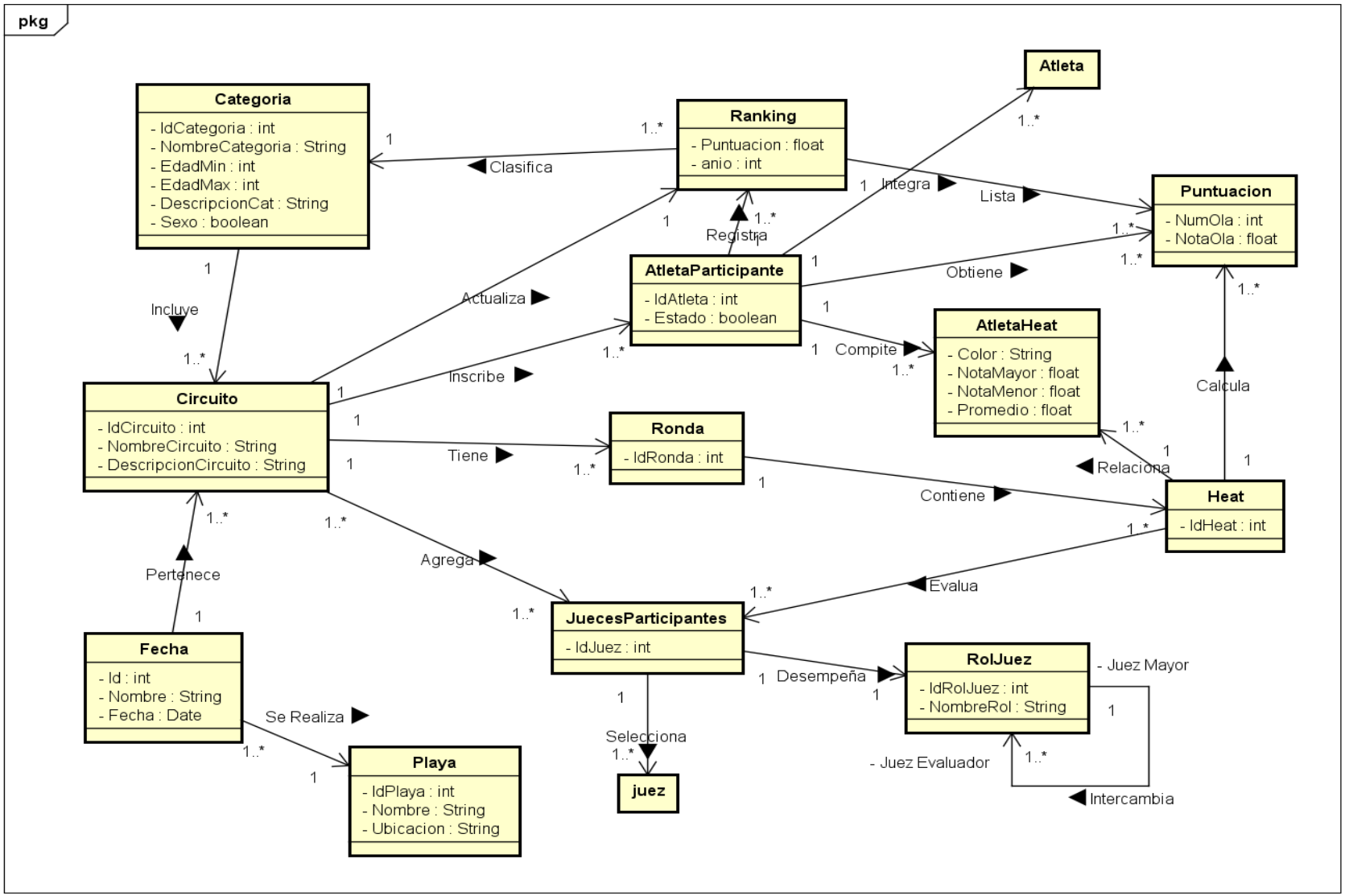
Identificación del requerimiento:	MOV02	
Nombre CU:	Buscar competencias.	
Objetivo:	El usuario deberá poder revisar todas las competencias que se realizan por medio de una búsqueda con parámetros.	
Descripción.	Se debe crear una pantalla de búsqueda donde se podrán ingresar como parámetros de entrada la fecha mínima y máxima de la búsqueda, la playa y estos resultados se ordenaran por fecha.	
Precondición.	El usuario debe tener instalada la aplicación	
Flujo de Éxito.	Paso	Acción
	1	El Usuario de la aplicación deberá seleccionar la opción de buscar competencias
	2	El usuario podrá ingresar o no parámetros de búsqueda
	3	El sistema mostrara todas las competencias que cumplan con los paramentos de la búsqueda.
Flujo Alternativo.	Paso	Acción
	3	Si no encuentra competencias , el sistema mostrara un mensaje de notificación de usuario
Pos-condición		
Frecuencia.	Cada vez que el usuario realice una búsqueda.	
Prioridad del CU:	Media	

Identificación del requerimiento:	MOV03	
Nombre CU:	Ver competencia.	
Objetivo:	El usuario podrá ver los detalles de una competencia	
Descripción.	Se debe crear una pantalla que permita mostrar todos los detalles de una competencia así como un top de sus participantes.	
Precondición.	El usuario debe previamente haber seleccionado una notificación o realizado una búsqueda	
Flujo de Éxito.	Paso	Acción
	1	El usuario selecciona una competencia ya sea por notificación o por búsqueda.
	2	El usuario podrá ver los datos de la competición
Flujo Alternativo.	Paso	Acción
Pos-condición		
Frecuencia.	Cada vez que un usuario realice una seleccione el resultado de una búsqueda o una notificación.	
Prioridad del CU:	Media	

Identificación del requerimiento:	MOV04	
Nombre CU:	Ver noticias.	
Objetivo:	El Usuario pueda al ingresar a la aplicación la lista de noticias más actuales	
Descripción.	Se debe crear una vista principal de la aplicación con las noticias y videos de la federación se deberá mostrar un nombre y una descripción breve de la noticia al presionar se deberá redirigir en navegador a un link configurado (podrá redirigir a videos de YouTube ,Facebook o a la página web de la federación)	
Precondición.	El usuario debe abrir la aplicación	
Flujo de Éxito.	Paso	Acción
	1	El usuario inicia la aplicación móvil
	2	El usuario podrá seleccionar una de las noticias mostradas
	3	El sistema deberá dirigir a la noticia
Flujo Alternativo.	Paso	Acción
Pos-condición		
Frecuencia.	Cada vez que se inicia el sistema	
Prioridad del CU:	Media	

2.5.2. Modelo de Dominio

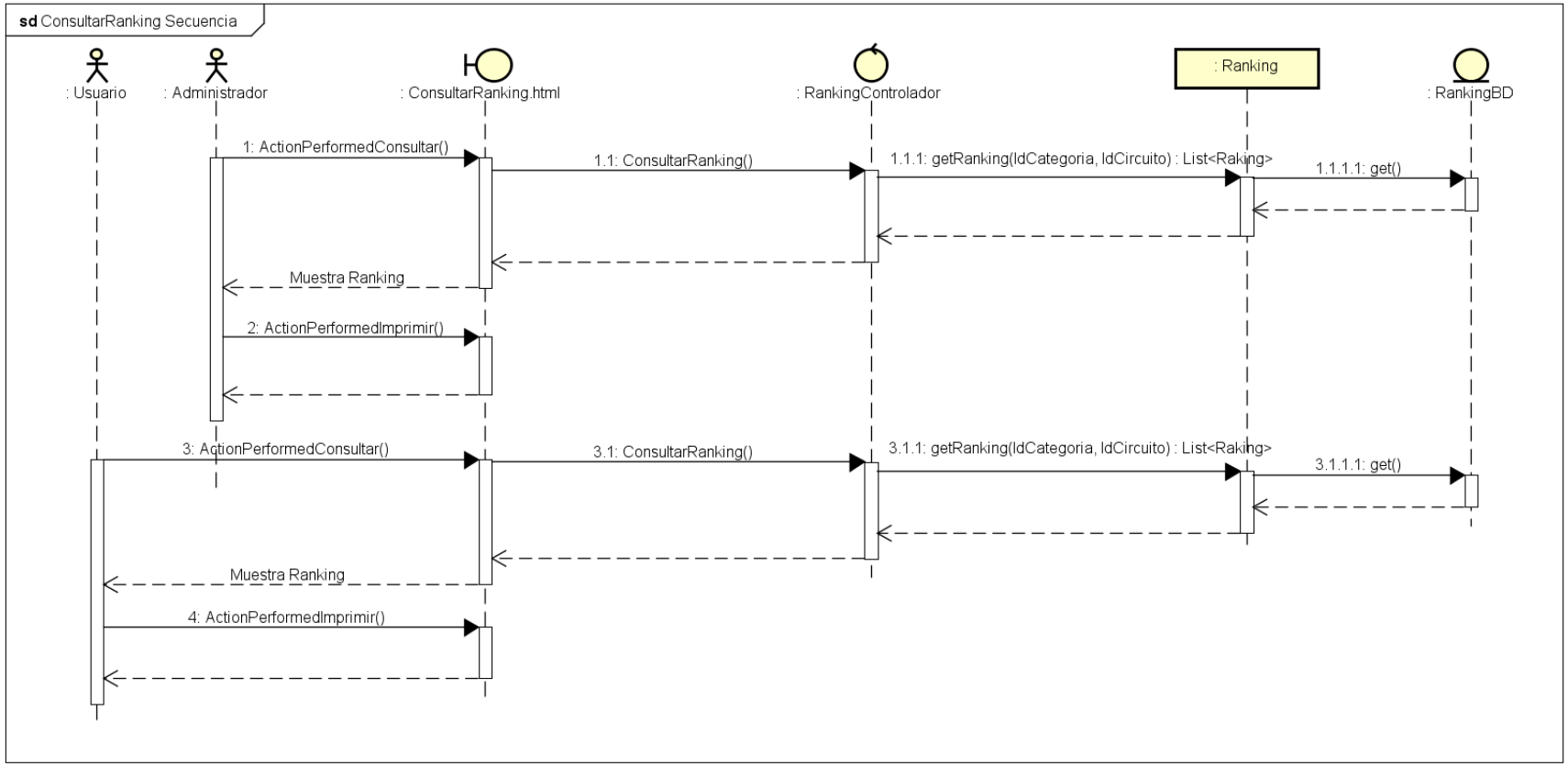
De los diferentes módulos principales del sistema a continuación se muestra el modelo de dominio del **Módulo de Competencias.**



2.5.3. Diagramas de Secuencia

De los diferentes módulos principales del sistema a continuación se muestra el diagrama de secuencia **Consultar Ranking** del

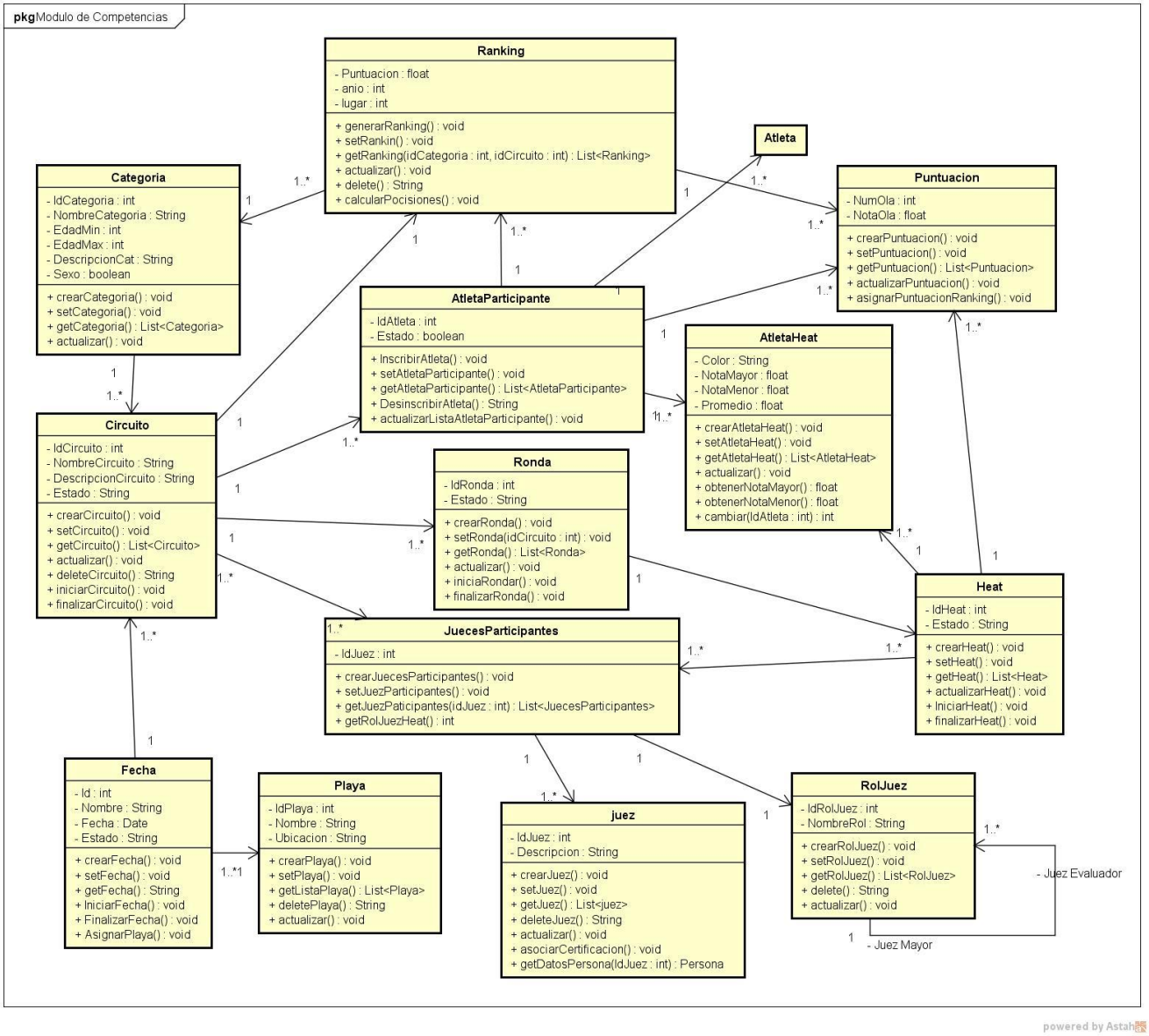
Módulo de Competencias.



2.5.4. Diagrama de Clases.

De los diferentes módulos principales del sistema a continuación se muestra el diagrama de clases del

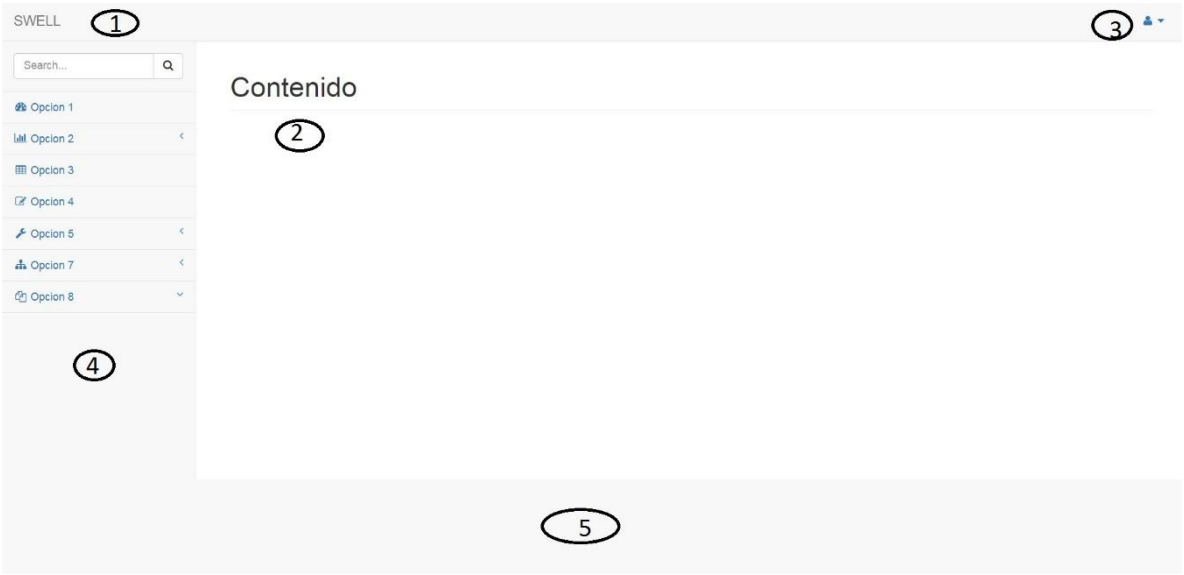
Módulo de Competencias.



3. CAPITULO III: DISEÑO DEL SISTEMA

3.1. Estándares de Diseños

3.1.1. Estándares para Pantalla Principal



Numero	Nombre	Descripción
1	Nombre	Nombre del sistema
2	Contenido	Tendrá el contenido de las páginas y este varía según cada pantalla
3	Menú Usuario	Contiene las opciones para el cambio de contraseña y cierre de sesión
4	Menú	Navegabilidad del sistema el menú se genera dinámicamente dependiendo el usuario que acceda al sistema.
5	Pie	Presentará información sobre la federación de surf

3.1.2. Estándares para Pantallas de Entrada

Numero	Descripción	Formato	Ejemplo
1	Representación de moneda	99999.99	Para representar las cantidades de dinero, se presentará el signo de dólar como prefijo en el caso de visualización de datos para entrada se añadirá automáticamente
2	Representación de las fechas	99/99/9999	El formato adoptado para poder ingresar y presentar las fechas es: DD/MM/YYYY el cual indica el día, mes y año.
3	Representación de enteros	9999999	Se utilizará los enteros para representar cantidades tales como edad, numero de Heat, número de participantes.
4	Representación Textos	xx-nn-xx	Para representar las cadenas de caracteres se utilizará la letra “x” intercalando la longitud de la cadena, será utilizado en descripciones, nombres.

3.1.3. Estándares para Pantallas de Salida

Numero	Nombre	Descripción
1	Opciones de Tabla	Número de líneas que llevará cada salida.
2	Buscador	Este buscará en la información que se presenta en la tabla.
3	Tabla	Contendrá el resultado de la información solicitada por el usuario.
4	Paginado	Según la cantidad de información que se obtenga se ordenará por páginas con un número de líneas definidas.

3.1.4. Estándares para Reportes

Convención de nombres y formato

Elemento	**Descripción**	**Ejemplo**
Nombre de reporte	El nombre del reporte debe ir al centro en la parte superior y debe describir el objetivo del reporte.	-Ranking Competencia -Atletas Inscritos
Identificador de reporte	El identificador de reporte se conformará por la siguiente estructura: “RP_”, seguido de módulo del sistema al que pertenece el reporte, seguido de “_” y luego un correlativo de reporte.	RP_PER_001 RP_IG_001 RP_COM_001
Formato de contenido	• El formato de letra será: Arial 11 • El color de la letra será negro y se ocupará el estilo negrito para resaltar aspectos como el tema del reporte y resultado de totales.	-Contenido **-Total**

Plantilla estándar para reportes que serán generados

LOGO	**Nombre de la Institución** **Nombre del Área** **Identificador de Reporte**	
Fecha: 99/99/9999	**Título de Reporte**	**Página 99** **De 999**
Variables Propias del Reporte		
Contenido del Reporte		
Nombre de Institución **Dirección de Institución**		

3.1.5. Estándares para Base de Datos

Elemento	Descripción	Ejemplo
Nombre de Esquema	El nombre de esquema utilizará una palabra definida en minúscula.	-atleta -circuito -categoría
Nombre de Tabla	El nombre de Tabla utilizará la siguiente nomenclatura: en minúscula cada palabra, en caso de un nombre con dos o más palabras, éstas serán separadas por un guion bajo "_". Cada palabra deberá ser singular.	-atleta_heat -rol -juez_heat
Nombre de Campo de Tabla	El nombre debe utilizar la siguiente nomenclatura: cada palabra será escrita en minúscula, en caso de un nombre con dos o más palabras, éstas serán separadas por un guion bajo "_" y podrán ser singular o plural.	-otros_estudios -id -descripción
Nombre de llaves primarias	El nombre de la llave primaria utilizara la siguiente nomenclatura: el nombre de la Tabla, antecedido de "id_".	id_atleta id_circuito
Nombre de llaves foráneas	El nombre de la llave foránea utilizará la nomenclatura: "fk_", seguido el nombre del id referenciado en notación lowerCamelCase, ambos separados por un guion bajo "_".	fk_id_heat fk_id_club
Nombre de Función	El nombre de funciones utilizará la nomenclatura: un nombre descriptivo en notación lowerCamelCase, antecedido de "f_".	f_calcularPuntos () f_inscribirAtleta ()

Nombre de Procedimiento	El nombre de Procedimientos utilizará la nomenclatura: un nombre descriptivo en notación lowerCamelCase, antecedido de “p_”.	p_generarHeats ()
Nombre de Trigger	El nombre de Triggers utilizará la nomenclatura: un nombre descriptivo en notación lowerCamelCase, antecedido de “tr_”.	tr_registrarTransacciones ()
Nombre de Vista	El nombre de Vistas utilizará la nomenclatura: un nombre descriptivo en notación lowerCamelCase, antecedido de “v_”.	v_circuitoRonda

3.1.6. Estándares de Programación

Los estándares utilizados que se adoptaron de las tecnologías involucradas para el desarrollo del sistema SWELL, fueron:

- Estándares implementados en Vue JS, Framework para facilitar el manejo de componentes de vistas.
    - Estándares de Programación JavaScript
    - Estándares CSS3 y HTML5.
- Estándares implementados en el lenguaje de programación JavaScript
    - Especificaciones y Convenciones de nomenclatura ECMAScript
        - Sentencias
        - Constantes
        - Métodos
        - Tipos
        - Let Variables
        - Componentes
        - Archivos
        - Repeticiones
        - Condicionales
        - Comentarios
        - Clases
        - Promesas
    - Especificaciones ECMAScript para React Native (App Móvil)

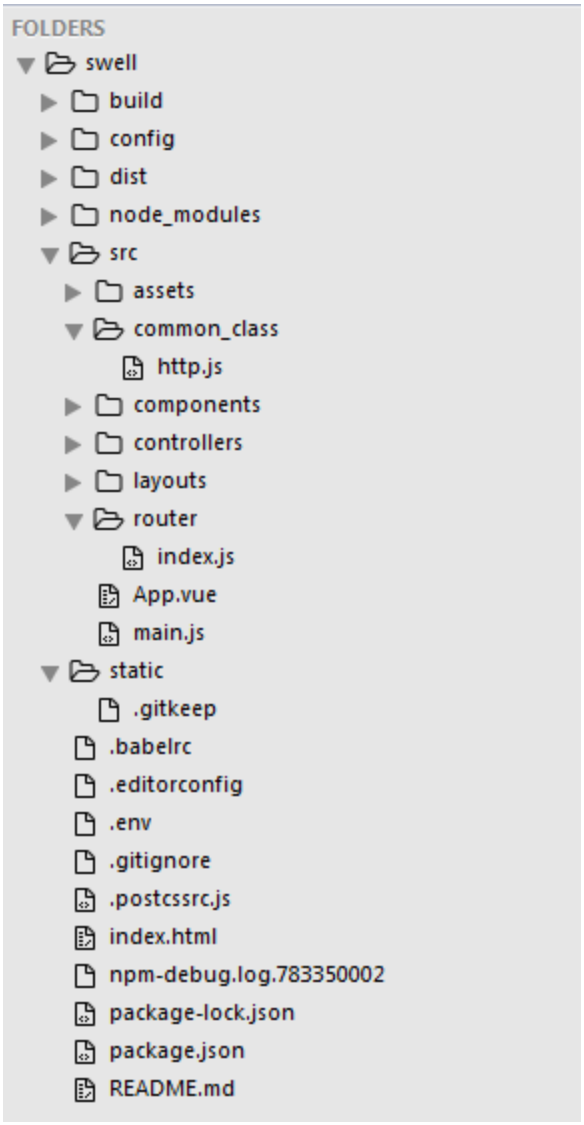
- Estándares implementados en el lenguaje de programación JAVA.
    - Nomenclatura de identificadores
    - Organización de archivos
    - Sentencias de código
    - Declaraciones de variables, clases e interfaces
    - Prácticas de programación

- Estándares implementados por Spring framework para desarrollo de aplicaciones web Java.
    - Patrón MVC
    - Inyección de Dependencias (ID)
    - Stack de tecnologías Java EE

- Servicios REST
  La publicación de un servicio proporcionado en REST requirió las mismas características que la publicación de un servicio SOAP, en cuanto a documentación propia del servicio (juego de pruebas, contrato de integración,...). Además se utilizó una serie de buenas prácticas para el correcto funcionamiento del servicio y la integración del mismo en las distintas aplicaciones que puedan consumirlo:
    - Uso de códigos de operación HTTP correctamente, evitando por ejemplo el uso del código GET para la inserción de un registro en lugar de utilizar POST para este fin.
    - Llamadas GET y HEAD sin alterar el estado de los registros o sin utilización para el transporte de datos sensibles.
    - Retorno de errores suficientemente claros y bien formados siguiendo la especificación lo más cercana al punto 5.2 Especificación Soap Fault para las llamadas erróneas que pudieran suceder.
    - Evitar uso de verbos en las URL, Uso de nombres y en plural.

3.1.6.1. Estándares de Organización de Ficheros

En Swell la configuración de archivos será de la siguiente manera:



En components están los archivos principales que integran los módulos en los cuales se ha trabajado en Swell, En esta se tendrá una carpeta por cada componente y por cada entidad que compone un módulo definido en Swell como atleta, transacción, playa, ronda, etc.

La configuración principal de Swell se encuentra en el archivo main.js El cual tiene la siguiente estructura básica:

```
// The Vue build version to load with the `import` command
// (runtime-only or standalone) has been set in webpack.base.conf with an alias.
import Vue from 'vue'
import App from './App'
import router from './router'
import * as uiv from 'uiv'
import moment from 'moment'
import {HTTP} from './common_class/http.js';
import esp from 'vee-validate/dist/locale/es';
import VueGoodTable from 'vue-good-table'
/* eslint-disable no-new */
import VeeValidate, { Validator } from 'vee-validate';
import VueMask from 'v-mask'
import VueFormWizard from 'vue-form-wizard'
import 'vue-form-wizard/dist/vue-form-wizard.min.css'
import VTooltip from 'v-tooltip'
import round from 'vue-round-filter';
import Toasted from 'vue-toasted';
import * as VueGoogleMaps from 'vue2-google-maps';
import FullCalendar from 'vue-full-calendar'
import "fullcalendar/dist/fullcalendar.min.css";
import  Tweet  from 'vue-tweet-embed';
Vue.use(FullCalendar)

Vue.config.productionTip = false;

Vue.use(VueGoogleMaps, {
  load: {
    key: 'AIzaSyDRuXnqROzf88kyefcZVSlhPkapyknqUTM',
    libraries: 'places', // This is required if you use the Autocomplete plugin
    // OR: libraries: 'places,drawing'
    // OR: libraries: 'places,drawing,visualization'
```

Convenciones de nomenclatura

Convenciones de nomenclatura generales

1. **El nombre de paquetes debería estar totalmente en minúsculas**. Esto se basa en la convención especificada por Sun para la nomenclatura de paquetes.

   ```
   mypackage, com.company.application.ui
   ```

   El nombre inicial de dominio del paquete debe estar totalmente en minúsculas.

2. **Los nombres que representan tipos, deben ser sustantivos y deben escribirse con mayúsculas y minúsculas iniciando con mayúscula.**

   ```
   Line, AudioSystem
   ```

   Esta es una práctica común en la comunidad de desarrollo en Java e incluso la convención de nomenclatura de tipos que utiliza Sun en los paquetes predefinidos de Java.

3. **Los nombres de variables deben utilizar mayúsculas y minúsculas, iniciando con minúscula.**

```
line, audioSystem
```

Es una práctica común en la comunidad de desarrollo Java y también la convención de nomenclatura de nombres de variables utilizada por Sun para los paquetes predefinidos de Java. Esta hace que las variables sean fáciles de distinguir de los tipos y resuelve de manera efectiva las posibles colisiones de nombre como en la declaración `Line`

```
line;
```

4. **Los nombres que representan constantes (variables finales) deben estar totalmente en mayúsculas utilizando subrayados (guión bajo) para separar las palabras.**

```
MAX_ITERATIONS, COLOR_RED
```

Es una práctica común en la comunidad de desarrollo en Java y también la convención de nomenclatura de nombres de utilizada por Sun para los paquetes predefinidos de Java. En general, debería minimizarse el uso de dichas constantes. En muchos casos es una mejor opción implementar el valor como un método.

```
int getMaxIterations() // NOT: MAX_ITERATIONS = 25
{
  return 25;
}
```

Esta forma es más fácil de leer y asegura una interface uniforme hacia los valores de la clase.

5. **Los nombres que representan métodos deben ser verbos y escribirse con mayúsculas y minúsculas iniciando con minúscula.**

```
getName(), computeTotalWidth()
```

Es una práctica común en la comunidad de desarrollo en Java y también la convención de nomenclatura de nombres de utilizada por Sun para los paquetes predefinidos de Java. Es idéntico al caso de los nombres de variables, pero los métodos en Java ya se distinguen de las variables por su forma particular.

6. **Las abreviaturas y acrónimos no deberían estar con mayúsculas cuando se usan como nombre.**

```
exportHtmlSource();// NOT: exportHTMLSource();
openDvdPlayer();   // NOT: openDVDPlayer();
```

Utilizar todo en mayúsculas para el nombre base entraría en conflicto con las convenciones anteriores. Una variable de este tipo debería nombrarse dVD, hTML, etc., lo cual obviamente no es legible.

Otro problema se ilustra en el ejemplo anterior. Cuando el nombre se conecta a otro, se reduce seriamente la legibilidad. La palabra a continuación del acrónimo no queda como debería.

7. **Variables privadas de clase**. Las variables privadas de clase deberían tener como prefijo un guión bajo (_).

```
class Person
{
  private String _name;
  ...

}
```

Aparte de su nombre y tipo, el *ámbito* de una variable es su característica más importante. Indicar el ámbito de la clase utilizando guión bajo hace fácil distinguir las variables de clase de las variables descartables. Es importante dado que las variables de clase se considera que tengan un significado más alto que las variables de métodos, y deberían ser tratadas con especial cuidado por el programador.

Un efecto lateral de la convención de nombres con guión bajo es que resuelve el problema de encontrar nombres de variables para los métodos *set*.

```
void setName (String name)
  {
    _name = name;

  }
```

8. **Las variables genéricas deberían tener el mismo nombre que su tipo.**

```
void setTopic (Topic topic) // NOT: void setTopic (Topic value)

1. NOT: void setTopic (Topic aTopic)

2. NOT: void setTopic (Topic x)

void connect (Databasedatabase)

1. NOT: void connect (Database db)

2. NOT: void connect (Database oracleDB)
```

Esto reduce la complejidad al reducir el número de términos y nombres utilizados. También hace fácil deducir el tipo contando solo con el nombre de la variable. Si por alguna razón esta convención no parece *encajar*, esto es un fuerte indicador de que el nombre del tipo está mal elegido. Las variables no genéricas tienen un *rol*. Estas variables a menudo se pueden nombrar combinando el rol y el tipo.

```
PointstartingPoint, centerPoint;

NameloginName;
```

9. **Todos los nombres deberían escribirse en inglés**.

```
fileName; // NOT: filNavn
```

El inglés es el lenguaje preferido para el desarrollo internacional.

10. **Los nombres de variables con un ámbito amplio deberían tener nombres largos, las variables con ámbito pequeño deberían tener nombres pequeños.** Las variables descartables utilizadas para el almacenamiento temporal o índices es mejor que sean cortas. Un programador que lee dichas variables debería poder asumir que su valor no se utiliza más allá de unas pocas líneas de código. Las variables descartables comunes para los enteros son *i*, *j*, *k*, *m*, *n* y para los caracteres *c* y *d*.

11. **El nombre del objeto está implícito y debería evitarse en un nombre de método.** `line.getLength();`

```
// NOT: line.getLineLength();
```

El uso de nombre del objeto podría parecer natural, en la declaración de la clase, pero se vuelve superfluo al utilizarlo.

Convenciones de nomenclatura específicas

1. **Los términos *get/set* se deben utilizar cuando un atributo se accede directamente.**

```
employee.getName(); matrix.getElement (2, 4);
employee.setName (name); matrix.setElement (2,
4, value);
```

Esta es la convención de nomenclatura para los métodos de acceso y es utilizada por Sun en los paquetes predefinidos de Java. Cuando se escribe Java Beans esta convención en realidad es obligatoria para los métodos.

2. **El prefijo *is* debería usarse para las variables y métodos booleanos.**

```
isSet, isVisible, isFinished, isFound, isOpen
```

Esta es la convención de nomenclatura para las variables y métodos booleanos utilizada por Sun en los paquetes predefinidos de Java. Cuando se escribe Java Beans esta convención en realidad es obligatoria para los métodos.

El uso del prefijo *is* resuelve el problema de elegir mal los nombres booleanos como *status* o*flag*.*isStatus*o*isFlag* simplemente no encajan y el programador se ve forzado a elegir otro nombre con mayor sentido. Hay unas cuantas alternativas que encajan mejor que el prefijo *is* en algunas situaciones. Estos son los prefijos *has*, *can*, y *should*.

```
booleanhasLicense(); booleancanEvaluate();
booleanshouldAbort = false;
```

3. **El término *compute* se puede utilizar en métodos donde algo se calcula.**

```
valueSet.computeAverage();

matrix.computeInverse();
```

De al lector la impresión inmediata que es una posible operación de consumo de tiempo, y que si se utiliza de manera repetida, él debería considerar guardar el resultado. Un consistente uso del término mejora la legibilidad.

4. **El término *find* se puede utilizar en los métodos donde se está buscando algo.**

```
vertex.findNearestVertex();

matrix.findMinElement();
```

De al lector la impresión inmediata que este es un simple método de búsqueda con un mínimo de cálculo implicado. El uso consistente del término incrementa la legibilidad.

5. **El término *initialize* se puede utilizar donde se establece un concepto u objeto.**

```
printer.initializeFontSet();
```

El *initialize* del inglés americano debería utilizarse en vez del *initialise* del inglés británico. Debe evitarse la abreviación *init*.

6. **Las variables JFC (Java Swing) deberían tener como sufijo el tipo de elemento.**

```
widthScale, nameTextField, leftScrollbar, mainPanel, fileToggle, minLabel,

printerDialog
```

Esto mejora la legibilidad dado que el nombre da al usuario la impresión inmediata del tipo de variable y por consiguiente los recursos disponibles del objeto.

7. **La forma plural debería utilizarse en nombres que representen una colección de objetos.**

```
Collection points; // of Point int[] values;
```

Esto mejora la legibilidad, dado que el nombre da al usuario la idea inmediata del tipo de variable y las operaciones que se pueden realizar sobre el objeto.

8. **El prefijo *n* debería utilizarse para variables que representen un número de objetos.**

```
nPoints, nLines
```

La notación es tomada de las matemáticas, donde es una convención para indicar un número de objetos. Note que Sun utiliza el prefijo *num* en los paquetes predefinidos de Java para tales variables. Esto probablemente debe significar una abreviación de *numberof*, pero más parece *number* lo que hace ver a la variable extraña y engañosa. Si "numberof" es la sentencia preferida, entonces debería usarse *numberOf* en vez de simplemente *n*. El prefijo *num*no debe usarse.

9. **El sufijo *No* debería utilizarse para variables que representen números de entidad.**

```
tableNo, employeeNo
```

La notación se toma en las matemáticas donde hay una convención establecida para indicar un número de entidad. Una alternativa elegante es colocar *i* como prefijo en dichas variables: *iTable*, *iEmployee*. Esto los hace de manera efectiva iteradores *nombrados*.

10. **Las variables Iterator deberían llamarse *i,j,k*, etc.**

```
while (Iterator i = points.iterator(); i.hasNext(); ) {

  :

}

for (inti = 0; i<nTables; i++) {

  :

}
```

Esta notación es tomada de las matemáticas donde hay una convención establecida para indicar a los iteradores. Las variables con nombre *j*, *k*, etc. deberían usarse solo para iteraciones anidadas.

11. **Los nombres complementarios deben usarse para las entidades complementarias**.

```
get/set, add/remove, create/destroy, start/stop, insert/delete,
increment/decrement, old/new, begin/end, first/last, up/down,
min/max,next/previous, old/new, open/close, show/hide
```

Esto reduce la complejidad en base a la simetría.

12. **Deben evitarse las abreviaciones en los nombres.**

```
computeAverage(); // NOT: compAvg(); ActionEvent event; // NOT:
ActionEvent e;
```

Hay dos tipos de palabras a considerar. Primero están los tipos de palabras comunes listadas en un diccionario de idiomas. Estas nunca deben ser abreviadas. Nunca escriba:

```
          en vez
cmd       de       command
          en vez
comp      de       compute
```

```
          en vez
cp        de       copy
          en vez
e         de       exception
          en vez
init      de       initialize
          en vez
pt        de       point
etc.
```

Por otro lado hay frases específicas difundidas que se conocen más por su acrónimo o abreviación. Estas frases deberían mantenerse abreviadas.

```
                          en vez
HypertextMarkupLanguage   de       html
                          en
CentralProcessingUnit     vez de   cpu
                          en
PriceEarningRatio         vez de   pe
```

etc.

13. **Las variables booleanas negadas deben evitarse**.

```
booleanisError; // NOT: isNotErrorbooleanisFound; // NOT:
isNotFound
```

El problema surge cuando el operador lógico *no* se usa y surge una doble negación. No es muy claro inicialmente el significado de !isNotError.

14. **Las constantes (variables finales) asociadas deben prefijarse con un nombre de tipo común.**

```
final int    COLOR_RED = 1;
              COLOR_GREEN =
final    int             2;
final    int COLOR_BLUE= 3;
```

Esto indica que las constantes se relacionan y el concepto que las constantes representan. Una alternativa a esta propuesta es poner las constantes dentro de una interfaz y así prefijar sus nombres con el nombre de la interfaz.

```
interface Color
{
    final int RED    = 1;
     final int GREEN = 2;
    final int BLUE   = 3;
}
```

15. **Las clases Exception deberían tener como sufijo*Exception*.**

```
classAccessException

{

  :

}
```

Las clases Exception no son realmente parte del diseño principal del programa y nombrarlas así las hace ser independientes con respecto a otras clases. Este estándar es seguido por Sun en la biblioteca básica de Java.

16. **Las implementaciones por defecto de las interfaces deben tener como prefijo *Default*.**

```
  classDefaultTableCellRenderer implements
  TableCellRenderer

{

  :

}
```

No es poco común crear una clase de implementación simplista de una interfaz, la cual proporcione comportamiento por defecto para los métodos de la interfaz. La convención de poner el prefijo *Default* a estas clases ha sido adoptada por Sun para las bibliotecas de Java.

17. **Las funciones (métodos que retornan un objeto) deberían nombrarse después de lo que retornan y los procedimientos (métodos *void*) después de lo que hacen.**

Esto incrementa la legibilidad. Aclara qué es lo que la unidad debería hacer y todas las cosas que *no* se supone que hace. Esto nuevamente facilita el mantener el código limpio de efectos laterales.

Archivos

Convenciones generales

1- **Los archivos fuente de Java deberían tener la extensión.*java*.**

```
Point.java
```

Esto es reforzado por las herramientas de Java.

2. **Las clases deberían declararse en archivos individuales con un nombre de archivo que concuerde con el nombre de la clase.** Las clases privadas secundarias pueden declararse como clases internas y residir en el archivo de la clase a la que pertenecen. Esto es apoyado por las herramientas de Java.
3. **El contenido del archivo debe mantenerse en 80 columnas.** 80 columnas es la dimensión común para los editores, emuladores de terminal, impresoras y depuradores; los archivos que se comparten entre varios desarrolladores deberían alinearse a estas restricciones. Cuando el archivo pasa por varios programadores, se mejora la legibilidad cuando se evitan los cortes de línea sin intención.
4. **Debe evitarse caracteres especiales como TAB y salto de página.** Estas características tienden a causar problemas en los editores, impresoras, emuladores de terminales o depuradores cuando se usan en entornos multi-plataforma y/o multi-programador.

5. **Debe evidenciarse cuando una línea está incompleta por estar partida.**

```
totalSum = a + b + c + d
+ e;
```

```
function (param1, param2,
param3);
```

```
setText ("Long line split" + "into
two parts.");
```

```
for (tableNo = 0; tableNo<maxTable; tableNo +=
tableStep)
```

Particionar líneas ocurre cuando una sentencia ocupa más del límite de 80 columnas especificado arriba. Es difícil dar reglas rígidas de cómo deberían partirse las líneas. Pero los ejemplos pueden dar una pista general.

1. Corte después de una coma.
2. Corte después de un operador.
3. Alinear la nueva línea con el inicio de la expresión en la línea anterior.

Sentencias

Sentencias package e import

1. **La sentencia package debe ser la primera sentencia del archivo**. Todos los archivos pertenecen a un paquete específico. La ubicación de la sentencia package es apoyada por el lenguaje Java. El permitir que todos los archivos pertenezcan a un paquete real (en vez del Java por defecto) refuerza las técnicas de programación orientadas a objeto de Java.

2. **La sentencia import debe seguir a las sentencia package**. La sentencias import deben ordenarse colocando el paquete fundamental primero, y agrupando los paquetes asociados, y colocando una línea en blanco de separación entre grupos.

```
import java.io.*; import java.net.*;

import java.rmi.*

import java.rmi.server.*;

import javax.swing.*;

import javax.swing.event.*;

import org.linux.apache.server.*;
```

   La ubicación de la sentencia import es apoyada por el lenguaje Java. El ordenamiento facilita listar cuando hay muchos import, y hace fácil de determinar las dependencias de los paquetes presentes. El agrupamiento reduce la complejidad contrayendo la información relacionada en una unidad común.

Clases e interfaces

1. **Las declaraciones de clase deberían organizarse.** Esto debería hacerse de la siguiente manera:

1. Documentación de la Clase/Interface.
2. Sentencia class o interface.
3. Variables de clase (estáticas) en el orden public, protected, package (sin modificador de acceso), privadas.
4. Variables de instancia en el orden public, protected, package (sin modificador de acceso), private.
5. Constructores.
6. Métodos (sin orden específico).

   Reduzca la complejidad haciendo predecible la ubicación de cada elemento de clase.

Métodos

1. **Los modificadores de métodos deberían darse en el siguiente orden: <acceso>static abstract synchronized<inusual> final native.** El modificador <acceso> (siexiste) debe ser el primer modificador.

   <acceso> es *public*, *protected*, o *private* mientras que <inusual> incluye *volatile* y *transient*. Lo más importante aquí es mantener el modificador *acceso* como el primer modificador. De entre los posibles modificadores, este es de lejos el más importante y debe estar al inicio en la declaración del método. Para otros modificadores, el orden es menos importante, pero es conveniente tener una convención fija.

Tipos

1. **Las conversiones de tipos deben hacerse siempre explícitas.** Nunca caiga en conversiones implícitas.

   ```
   Float Value = (float) int Value; // NOT: float Value = int Value;
   ```

   El programador debe indicar que está consciente de los diferentes tipos implicados y que la mezcla es intencional.

Variables

1. **Las variables deberían inicializarse donde se declaran y deberían declararse en el ámbito más pequeño posible.** Esto asegura que las variables son válidas en todo momento. A veces es imposible inicializar una variable con un valor válido donde se le declara. En esos casos debería dejarse sin inicializar en vez de darle un valor sin sentido.

2. **Las variables nunca deben tener significado dual.** Esto mejora la legibilidad asegurando que todos los conceptos se representan de manera única. Reduce las posibilidades de errores producto de la dualidad.

3. **Las variables de clase nunca se deberían declarar public.** El concepto de ocultamiento de información y encapsulamiento de Java es violado por las variables públicas. Use las variables privadas y acceda en vez de ello a las funciones. Una excepción a esta regla es cuando la clase es básicamente una estructura de datos, sin comportamiento (equivalente al struct de C++). En este caso es apropiado hacer las variables de instancia de la "clase" públicas.

4. **Las variables relacionadas del mismo tipo se pueden declarar en una sentencia común.** Las variables no relacionadas no se deberían declarar en la misma sentencia.

   ```
   Float x, y, z;

   Float revenueJanuary, revenueFebrury, revenueMarch;
   ```

   El requerimiento común de tener declaraciones en líneas separadas no es útil en los ejemplos. Esto mejora la legibilidad al agrupar variables. Note que sin embargo que cuando sea posible, las variables deberían inicializarse donde se declaren, en caso que esta situación no ocurra.

5. **Los arreglos deberían declararse con corchetes junto al tipo**.

```
double[]     vertex;// NOT: double vertex[];
int[]        count; // NOT: int    count[];


           public static void main (String[] arguments)

           public double[] computeVertex()
```

Esto tiene doble motivo. Primero, la característica de ser arreglo es de la clase, no de la variable. Segundo, cuando se retorna un arreglo en un método, no es posible tener los corchetes en otro lugar que no sea con el tipo.

6. **Las variables deberían mantenerse con vida el menor tiempo posible**. Mantenerlas operaciones sobre una variable dentro de un ámbito pequeño, es más fácil de controlar efectos y efectos laterales de la variable.

Repeticiones

1. **Sólo las sentencias de control de flujo deben incluirse en la construcción del *for()***.

```
sum = 0;



for (i = 0; i< 100; i++) sum += value[i];
```

1. `NOT: for (i=0, sum = 0; i< 100; i++)`
2. `sum += value[i];`

Esto incrementa la legibilidad y mantenibilidad. Hace una clara distinción que *controla* y que está *contenida* en la repetición.

2. **Las variables de repetición deberían inicializarse inmediatamente antes de la repetición.**

```
                              NOT: isDone =
    booleanisDone = false; // false;
while (!isDone) {          //     :
                              while (!isDone)
      :                    // {
}                          // :

              }
```

3. **Debería evitarse el uso de ciclos do…while.**
   Hay dos razones para esto. Primero es que la construcción es superflua. Cualquier sentencia que se puede escribir como un *do…while* se puede igualmente escribir como una repetición *while* o una repetición *for*. La complejidad se reduce si se utiliza la menor cantidad de construcciones. La otra razón es legibilidad. Un ciclo con la parte condicional el final es más difícil de leer que uno con el condicional arriba.

4. **Se debería evitar el uso de *break* y *continue* en las repeticiones.**
   Estas sentencias sólo se deberían usar si demuestran brindar más legibilidad que sus contrapartes estructuradas.

Condicionales

1. Se debe evitar las expresiones condicionales.

Introduzca en su lugar variables booleanas temporales.

```
        if ((elementNo< 0) || (elementNo>maxElement)|| elementNo ==
        lastElement) {
      :
}
```

Debería reemplazarse por:

```
booleanisFinished = elementNo< 0 || elementNo>maxElement; booleanisRepeatedEntry
= elementNo == lastElement;
if (isFinished || isRepeatedEntry) {
  :
}
```

Al asignar variables booleanas a las expresiones, el programa obtiene documentación automáticamente. La construcción será más fácil de leer, depurar y mantener.

2. **El caso nominal debería colocarse en la parte *if* y la excepción en la parte *else* de una sentencia if.**

```
booleanisError = readFile (fileName);
if (!isError) {
  :
}
else {
  :
}
```

Asegúrese que las excepciones no restan claridad a la ruta normal de ejecución. Esto es importante tanto para la legibilidad como para el rendimiento.

3. El condicional debería colocarse en una línea separada.

```
if (isDone) // NOT: if (isDone) doCleanup(); doCleanup();
```

Esto es para propósitos de depuración. Cuando se escribe en una sola línea no es visible cuando la evaluación es verdadera o falsa.

4. Debe evitarse las sentencias ejecutables en los condicionales.

```
file = openFile (fileName); // NOT: if ((file = openFile (fileName)) != null) {
if (file != null) {             //     :
        :                       //    }
}
```

Los condicionales con sentencias ejecutables son difíciles de leer. Esto es particularmente cierto para los programadores en Java.

Miscelánea

1. **El uso de números mágicos en el código debería evitarse.** Los números diferentes de *0* y *1* se pueden considerar en vez de eso, declarados como constantes nombradas.

```
private static final int TEAM_SIZE = 11;

:

Player[] players = new Player[TEAM_SIZE]; // NOT:
Player[] players = new Player[11];
```

Si el número no tiene un significado obvio de por si, la legibilidad se mejora introduciendo en vez de eso una constante nombrada.

2. **Las constantes de punto flotante siempre se deberían escribir con punto decimal y con al menos un decimal.**

```
double total = 0.0; // NOT: double total = 0; double speed =
3.0e8; // NOT: double speed = 3e8;

double sum;

:


sum = (a + b) * 10.0;
```

Esto enfatiza la naturaleza diferente de los números enteros y de punto flotante. Matemáticamente los dos modelan conceptos diferentes y no compatibles. También, como en el ejemplo, esto enfatiza el tipo de variable asignada (sum) al punto en el código donde no podría ser evidente.

3. **Las constantes de punto flotante siempre se deberían escribir con un dígito antes del punto decimal.**

```
double total = 0.5; // NOT: double total = .5;
```

El número y sistema de expresión en Java es tomado de las convenciones matemáticas para la sintaxis en lo posible. Asimismo, 0.5 es más legible que .5; No hay manera que se pueda confundir con el entero 5.

4. **Las variables o métodos estáticos siempre se deben referir por el nombre de la clase y nunca por el nombre de una instancia.**

```
Thread.sleep (1000); // NOT: thread.sleep (1000);
```

Esto enfatiza que el elemento que se referencia es estático e independiente de cualquier instancia particular. Por la misma razón, el nombre de clase debería incluirse cuando una variable o método se accede desde dentro de la misma clase.

Organización y comentarios

Organización

1. **La indentación básica debería ser 2.**

```
for (i = 0; i<nElements; i++) a[i] = 0;
```

La indentación se utiliza para enfatizar la estructura lógica del código. La indentación de 1 es muy pequeña para alcanzarlo. La indentación mayor a 4 hace que el código muy profundo y difícil de leer e incrementa la probabilidad de que las líneas se partan. Eligiendo entre las indentaciones de 2, 3 y 4; 2 y 4 son las más comunes, y 2 se elige para reducir las posibilidades de partir las líneas de código. Note que las recomendaciones de Sun en este punto son 4.

2. **Organización de Bloques.** La organización de bloques debería ser como se ilustra en el ejemplo 1 abajo (recomendado) o el ejemplo 2, pero no como se muestra en el ejemplo 3. Los bloques de clase, interfaz y método deberían usar la organización de bloques del ejemplo 2.

```
while (!isDone) {
  doSomething();
  isDone =
  moreToDo();
}
```

```
while (!isDone)
{
  doSomething();
            isDone =
moreToDo();
}
```

```
while (!isDone)
  {
    doSomething();
              isDone=
moreToDo();
  }
```

El ejemplo 3 introduce un nivel de indentación extra que no enfatiza la estructura lógica del código tan claramente como el ejemplo 1 y 2.

3. **Declaración de clase e interfaz.** Las declaraciones de clases e interfaces deberían ser de la siguiente forma.

```
      classSomeClass extends AnotherClass implements
      SomeInterface, AnotherInterface

    {

      ...

    }
```

Esto sigue la regla general de bloques especificada arriba. Note que es común en la comunidad de desarrolladores de Java tener llaves abiertas al final de la línea de la palabra reservada class. Esto no se recomienda.

4. **Declaración de métodos.** Las declaraciones de métodos deberían tener la siguiente forma.

```
      public void someMethod () throws
      SomeException

    {

      ...

    }
```

Vea los comentarios en las sentencias class arriba.

5. **If-else.** Las sentencias de clase if-else deberían tener la siguiente forma.

```
if (condition) {

    statements;
}

if (condition) {
    statements;
}
else {
    statements;
}

if (condition) {
    statements;
}
else if (condition) {
    statements;
}
else {
    statements;
}
```

Esto sigue en parte la regla general de bloques. Sin embargo, debería discutirse si la cláusula *else* debería estar en la misma línea que el cierre de corchetes de la cláusula *if* o *else* previa.

```
if (condition) {
    statements;
}
else if (condition) {
    statements;
}
else {
    statements;
}
```

Esto es equivalente a la recomendación de Sun. El estilo elegido se considera mejor en el sentido que cada parte de la sentencia if-else se escribe en líneas separadas del archivo. Esto haría más fácil manipular la sentencia, por ejemplo al mover las cláusulas *else*.

6. **Sentencia for.** La sentencia for debería tener la siguiente forma.

```
for (initialization; condition; update) { statements;

}
```

Esto concuerda con la regla general de bloques.

7. **Sentencia for vacía.** Una sentencia for vacía debería tener la siguiente forma.

```
for (initialization; condition; update)

  ;
```

Esto enfatiza el hecho de que la sentencia for está vacía y hace obvio para el lector que es intencional.

8. **Sentencia while.** La sentencia while debería tener la siguiente forma.

```
while (condition) { statements;



}
```

Esto sigue la regla general de bloques.

9. **Sentencia do-while.** La sentencia do-while debería tener la siguiente forma.

```
    do { statements;
} while (condition);
```

Esto sigue la regla general de bloques.

10. **Sentencia switch.** La sentencia switch debería tener la siguiente forma.

```
    switch (condition) { case ABC :
      statements;
      // Fallthrough case DEF :
      statements;
      break; case XYZ :
      statements;
      break; default :
      statements;
      break;
}
```

Esto difiere de la recomendación de Sun tanto para indentación como para espaciado. En particular, cada palabra reservada *case* se indenta relativa a la sentencia switch como un todo. Esto hace consistente toda la sentencia switch. Note igualmente el espacio extra antes del carácter. El comentario explícito *// Continuar ejecutando* debería incluirse cuando hay una sentencia case sin un cláusula *break.* El dejar la sentencia *break* fuera es un error común y debe quedar claro que es intencional cuando no está ahí.

11. **Sentencia try-catch**. Una sentencia try-catch debería tener la siguiente forma.

```
    try { statements;
}
    catch (Exception exception) { statements;
}
```

```
  try { statements;
}
  catch (Exception exception) { statements;
}
  finally { statements;
}
```

Esto forma parte de la regla general de bloques. Esta forma difiere de la recomendación de Sun en la misma forma que la sentencia *if-else* descrita arriba.

12. **Sentencias if-else, for o while simples.** Las sentencias if-else, for o while deberían escribirse sin llaves.

```
if (condition) statement;

while (condition) statement;

for (initialization; condition; update) statement;
```

Es una recomendación común (incluyendo la recomendación de Sun) que las llaves deberían siempre usarse en todos los casos. Sin embargo, las llaves son en general una construcción del lenguaje que agrupa varias sentencias. Las llaves son por definición superfluas en una sentencia simple. Un argumento contra esta sintaxis es que el código va a cortarse si se adiciona una sentencia sin agregar las llaves. En general, sin embargo, el código nunca debería escribirse para acomodar los cambios que podrían surgir.

Espacios en blanco

Reglas generales. Debería seguirse las siguientes reglas:

1. Los operadores deberían rodearse por un carácter de espacio.
2. Las palabras reservadas de Java deberían estar seguidas por un espacio en blanco.
3. Las comas deberían estar seguidas por un espacio en blanco.
4. Los dos puntos deberían estar rodeados por espacio en blanco.
5. Los puntos y coma en las sentencias *for* deberían estar seguidos de un espacio en blanco.

```
                *
a = (b + c)       d;              // NOT: a=(b+c)*d
while (true) {                    // NOT: while(true) ...
                                  // NOT: doSomething
doSomething (a,       b, c, d);   (a,b,c,d);
case 100        :                 // NOT: case 100:
                0; i  10; i++)    // NOT:
for (i =        <     {           for(i=0;i<10;i++){
```

Hace los componentes individuales de las sentencias consistentes y mejora la legibilidad. Es difícil dar una lista completa del uso sugerido del espacio en blanco en el código en Java. Los ejemplos arriba sin embargo deberían dar una idea general de las intenciones.

2. **Nombres de Funciones.** Los nombres de funciones deberían estar seguidos de un espacio en blanco cuando son seguidos de otro nombre.

```
                              NOT:
doSomething (parameter);    // doSomething(parameter);
doSomething();              // OK
```

Esto hace los nombres individuales consistentes y mejora la legibilidad. Cuando no lo sigue nombre alguno, el espacio puede omitirse dado que no hay duda sobre el nombre en este caso. Una alternativa a esta propuesta es requerir un espacio *después* de abrir el paréntesis. Los que se adhieren a este estándar usualmente dejan un espacio antes de cerrar el paréntesis. Esto hace los nombres individuales consistentes como es la intención, pero el espacio antes del cierre de paréntesis es algo artificial, y sin este espacio la sentencia luce asimétrica.

3. **Unidades lógicas.** Las unidades lógicas deberían separarse por una línea en blanco.

```
Matrix4x4 matrix = new Matrix4x4();

doublecosAngle = Math.cos (angle); double sinAngle =
Math.sin (angle);


matrix.setElement (1, 1, cosAngle); matrix.setElement
(1, 2, sinAngle); matrix.setElement (2, 1, -sinAngle);
matrix.setElement (2, 2, cosAngle);

multiply (matrix);
```

Esto mejora la legibilidad introduciendo un espacio en blanco entre las unidades lógicas de un bloque.

4. **Métodos.** Los métodos deberían separarse por 3-5 espacios en blanco. Haciendo el espacio mayor que el espacio dentro de un método, los métodos serán diferenciados dentro de la clase.

5. **Variables.** Las variables en las declaraciones deberían alinearse a la izquierda, mejorando así la legibilidad del código.

```
TextFile file;

intnPoints;

doublex, y;
```

Las variables son más fáciles de distinguir de los tipos alineándolas.

6. **Sentencias.** Las sentencias deberían alinearse donde se mejore la legibilidad.

```
if          (a == lowValue)
               compueSomething();
elseif      (a == mediumValue)
               computeSomethingElse();
elseif      (a == highValue)
            computeSomethingElseYet();
value =     (potential* oilDensity)       / constant1 +
            (depth  * waterDensity)        / constant2 +
            (zCoordinateValue * gasDensity)/ constant3;
                        minPosition = computeDistance (min, x, y, z);

averagePosition = computeDistance (average, x, y, z);


switch (value) {
  case PHASE_OIL    : phaseString = "Oil";           break;
  case  PHASE_WATER  : phaseString = "Water";        break;
  case PHASE_GAS ;  : phaseString = "Gas";           break;
}
```

Hay varios lugares en el código donde el espacio en blanco se puede incluir para mejorar la legibilidad incluso si esto viola las reglas comunes. Muchos de los casos tienen que ver con la alineación de código. Las reglas generales para la alineación de código son difíciles de dar, pero los ejemplos arriba deberían dar una idea general de la idea. En términos simples, cualquier construcción que mejore la legibilidad debería permitirse.

Comentarios

1. **Código confuso.** El código confuso no debería comentarse sino reescribirse. En general, el uso de los comentarios debería minimizarse haciendo el código auto documentado con elecciones apropiadas de nombres y estructuras lógicas explícitas.

2. **Idioma.** Todos los comentarios deberían escribirse en inglés. En un entorno internacional el inglés es el idioma preferido.

3. **Uso de //.** Use // para todos comentarios que no son de JavaDoc (convenciones de documentación de de Java), incluyendo los comentarios multilínea.

```
1. Comments panning
2. more tan one line
```

Dado que los comentarios multi-nivel no es soportado, el uso de los comentarios // asegura que siempre sea posible comentar secciones enteras de un archivo usando /* */ para propósitos de depuración, etc.

4. **Indentación de comentarios.** Los comentarios deberían indentarse relativos a suposición en el código.

```
                          // NOT: while
while (true) {            (true) {
                                // Do
  // Do something         //    something
  something();            //      something();
}                         //    }
```

Esto ayuda a evitar que los comentarios quiebren la estructura lógica del programa.

5. **Colecciones.** La declaración de variables *collection* debería estar seguida de un comentario que establezca el tipo común de los elementos de la colección.

```
private    Vector points_; // of Point
private    Set    shapes_; // of Shape
```

Sin el comentario extra puede ser difícil imaginar en qué consiste la colección y como tratar a los elementos de la colección. En métodos que toman como entrada variables colección, el tipo común de los elementos debería especificarse en el comentario asociado de JavaDoc.

6. **Clases y funciones.** Todas las clases públicas y las funciones públicas y protegidas dentro de las clases públicas deberían documentarse utilizando las convenciones de documentación de Java (Javadoc). Esto hace más fácil mantener al día la documentación en línea del código.

3.1.7. Estándares de documentación

Los estándares de documentación para el proyecto, están centrados en los siguientes aspectos:

Tipo de Letra

Elemento	Descripción	Ejemplo
Títulos	Los títulos de toda documentación reflejada en documentos definidos en este capítulo serán Arial y tamaño 14.	Circuito
Subtítulos	Los Subtítulos de toda documentación reflejada en documentos definidos en este capítulo, serán Arial y tamaño 12.	Categoría
Párrafo	Los párrafos de toda documentación reflejada en documentos definidos en este capítulo serán Arial y tamaño 11.	Atletas Participantes

Documentación interna del código del sistema

Elemento	Descripción	Ejemplo
Función o Modulo	Se debe documentar: -Descripción general de que hace la Función o modulo. -Tipo de parámetro recibido -Resultado devuelto	/* *Esta función Modifica un rol de un usuario *Recibe el nombre del rol, de tipo carácter *No retorna ningún valor a ser devuelto */ f_ModificarRol (char NombreRol)
Clase	Se debe documentar: -Autor de la clase	/* *Autor: Nombre del Programador ejemplo */ Class Atleta {…}

Archivo de configuración	Se debe documentar:
	-Descripción de variables de configuración
	-Autor del archivo
		/*
		*Autor: Nombre de quien configuro ejemplo
		*/
		// Nombre de la base de datos
		Database: FesasurfDB
		// Usuario de la base de datos
		User: Admin
		// Puerto
		Port: 3306

Manual de usuario

Este documento tiene como objetivo asistir al usuario en el uso correcto y que este pueda aprender todas las funcionalidades para la manipulación del sistema y dar solución a incidencias que puedan suceder en la operación de este.

Plantilla para documentar funcionalidades del sistema

<table>
<tr><td>LOGO</td><td>Nombre de la Institución
Nombre del Sistema
Nombre de Manual</td><td>Versión 99.99</td></tr>
<tr><td>Fecha: 99/99/9999</td><td>Tema de Contenido</td><td>Página 99
De 999</td></tr>
<tr><td colspan="3">Pantalla 99

Descripción pantalla 99</td></tr>
<tr><td colspan="3">Observaciones</td></tr>
<tr><td>Elaboro:</td><td>Reviso:</td><td>Aprobó:</td></tr>
</table>

Manual Técnico

Este documento tiene como objetivo instruir al administrador del sistema y a otros desarrolladores de software para que puedan proporcionar mantenimiento, si así fuese requerido.

Plantilla para documentar contenido

<table>
<tr><td>LOGO</td><td colspan="2">Nombre de la Institución
Nombre del Sistema
Nombre de Manual</td><td>Versión 99.99</td></tr>
<tr><td>Fecha: 99/99/9999</td><td colspan="2">Tema de Contenido</td><td>Página 99
De 999</td></tr>
<tr><td colspan="4">Contenido</td></tr>
<tr><td colspan="4">Observaciones</td></tr>
<tr><td>Elaboro:</td><td>Reviso:</td><td colspan="2">Aprobó:</td></tr>
</table>

Manual de Instalación/Desinstalación del software del sistema

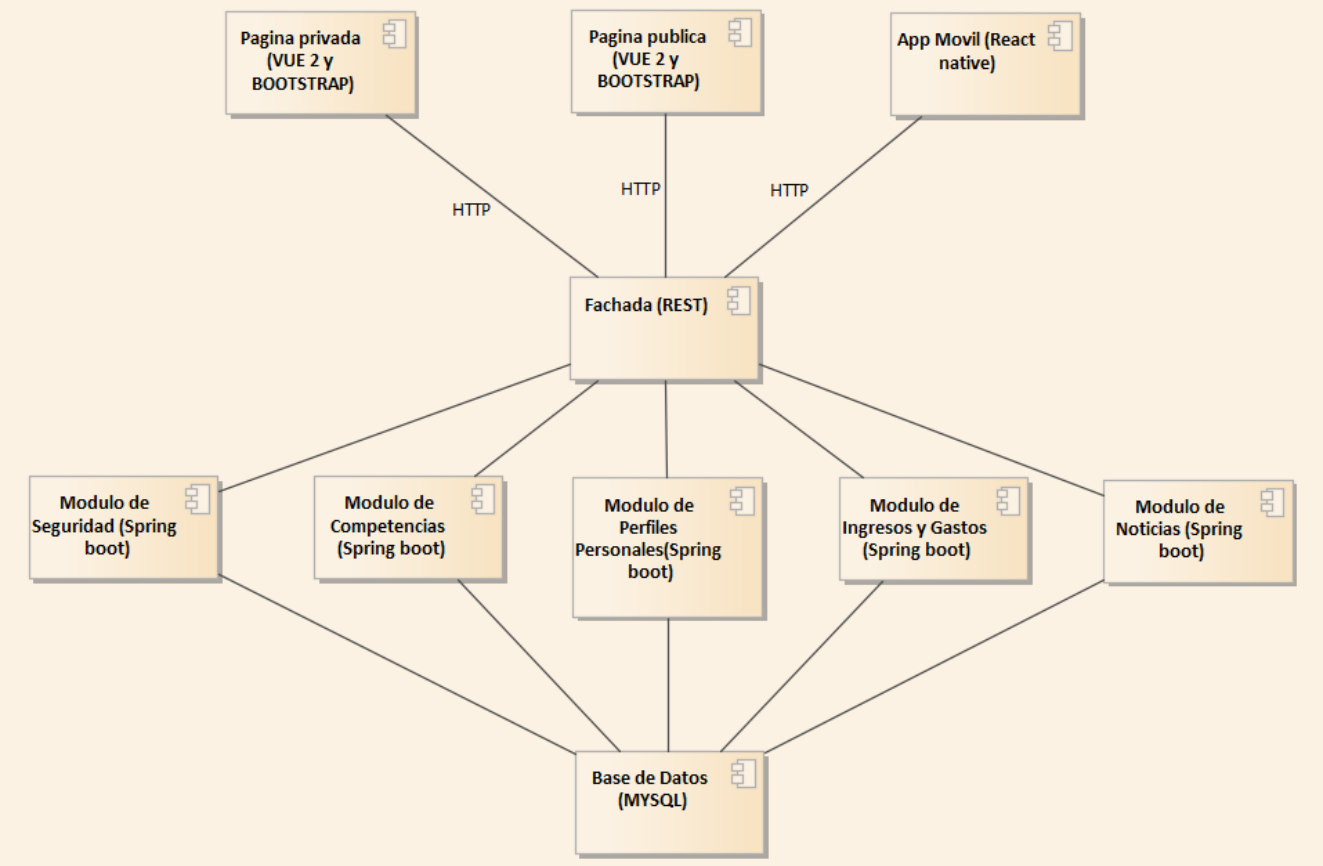
Este documento tiene como objetivo instruir al administrador los pasos a seguir para instalar/desinstalar correctamente el software del sistema.

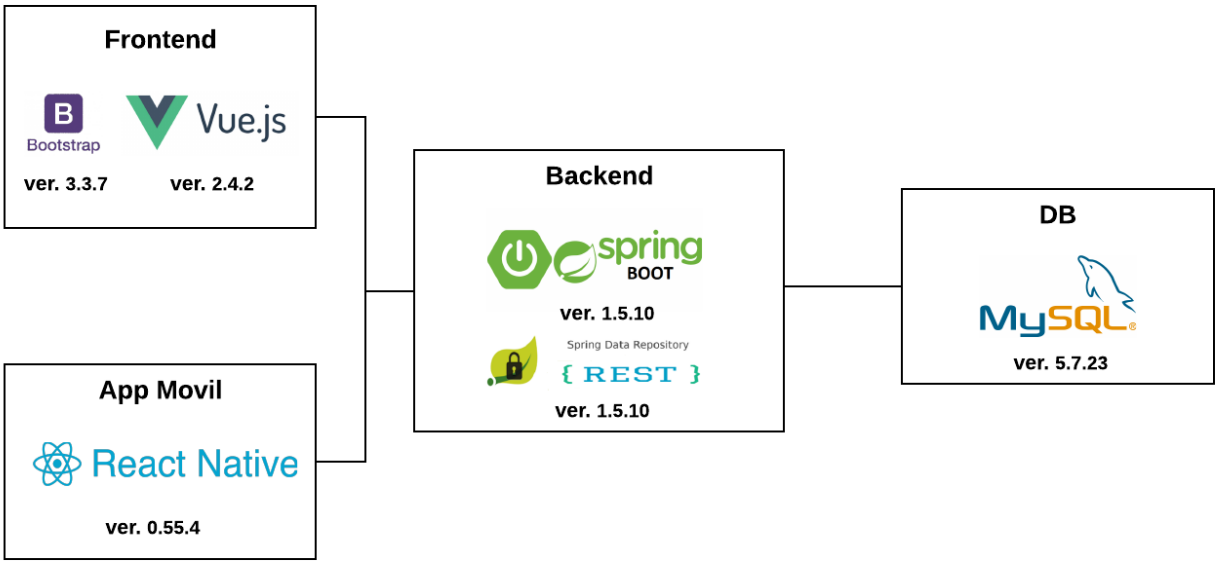
Plantilla para documentar proceso de instalación/desinstalación

LOGO	**Nombre de la Institución** **Nombre del Sistema** **Nombre de Manual**	**Versión 99.99**
Fecha: 99/99/9999	**Tema de Contenido**	**Página 99** **De 999**
Contenido		
Elaboro:	**Reviso:**	**Aprobó:**

3.2. Diseño Arquitectónico

A continuación, se presenta el diseño arquitectónico de Swell





3.2.1. Tecnologías utilizadas

Definición de tecnologías con las que se implementó la solución: FRONTEND

Plataforma de Desarrollo	JavaScript ES2015	JavaScript es un lenguaje de programación interpretado, dialecto del estándar ECMAScript. Se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico.
Lenguaje de Desarrollo	JS,HTML5,CSS3	JavaScript integrado con otros tipos de tecnologías como HTML5 and CSS3 que permiten mejorar el diseño de la aplicación
IDE	Indiferente	Atom, Visual Studio Code, etc.
Servidor	Apache 2	Es un servicio de páginas web HTTP de código abierto que sirve para colocar varias plataformas como Unix, BSD, GNU/Linux, Windows, Macintosh entre otros que implementan el protocolo HTTP
Framework	Vue JS 2.4.2	Vue.js es un framework de JavaScript que se enfoca principalmente en construir interfaces de usuario. Ya que solo se encarga de 'manipular' la capa de la vista puede ser integrado fácilmente con otras librerías.
Framework (App Móvil)	React Native 0.55.4	React Native es un framework desarrollado por Facebook que permite desarrollar apps nativas iOS y Android usando Javascript. Aplica tus conocimientos de JavaScript y React para crear una App de iOS y Android reutilizando el mismo código, manteniendo los componentes nativos para cada plataforma.
Esquema de presentación	Integración de Bootstrap 3.3.7, Vue JS y BackEnd con servicios REST	Esquema de presentación Web para los usuarios utilizando Bootstrap Integrado con BackEnd a través de Vue JS por medio de servicios REST

Definición de tecnologías con las que se implementó la solución: BACKEND

Plataforma de Desarrollo	Java 8	Java 8 es una de las versiones más reciente de Java que incluye nuevas características, mejoras y correcciones de bugs para mejorar la eficacia en el desarrollo y la ejecución de programas Java.
Lenguaje de Desarrollo	Java	Java es un lenguaje de programación de propósito general, concurrente, orientado a objetos, que fue diseñado específicamente para tener tan pocas dependencias de implementación como fuera posible.

IDE	STS(eclipse)	Eclipse es un entorno de desarrollo software multi-lenguaje construido alrededor de un workspace al que pueden incluirse un gran número de plug-ins que proporcionan funcionalidades concretas relacionadas con lenguajes específicos o con la interacción con otras herramientas implicadas en el desarrollo de una aplicación.
Base de Datos	MySQL 14.14 Distrib 5.7.23	MySQL es la base de datos de código abierto más popular del mundo. Con su rendimiento, confiabilidad y facilidad de uso comprobados, Para la Administración de la base de datos se hará uso de la herramienta MySql Workbench en su versión 6.2
Servidor	Apache Tomcat 1.5.10 Release	el servidor web usado tradicionalmente para proyectos Java por su implementación de servlets o páginas JSP
Framework	Spring boot 1.5.10 Release	Spring es un framework para el desarrollo de aplicaciones y contenedor de inversión de control, de código abierto para la plataforma Java.
Esquema de presentación	Spring Security y Repocitory 1.5.10 Release	Backend Rest elaborado en Spring con Spring Security y Repocitory para base de datos

3.3. Diseño de Base de Datos

Para el diseño de la base de datos se hiso uso de diferentes modelos como lo son: El modelo de Entidad Relación, Modelo lógico de datos y Modelo físico de datos

3.3.1. Diccionario de Datos

El diccionario de datos es un listado organizado de los datos que pertenecen a un Sistema, A continuación se muestra este listado para la tabla Atleta:

Nombre de la tabla	**Atleta**								
Descripción	Almacenar todos los datos generales de la entidad Atleta								
Nombre Campo	**Tipo dato**	**Formato**	**Tamaño**	**Dominio**	**Valor por Default**	**Formula**	**Nulo**	**Requerido**	**Descripción**
id	Entero	**99**	11		No Posee	No	No	Si	Identificador de cada registro, es generado de forma automática por el sistema
nivel_academico	Enum	**xxxx**	20	a-z A-Z	Sin Estudios	No	No	Si	Almacena el nivel académico que posee el atleta
u_anio_cursado	Caracter	**XXXX**	40	a-z A-Z	Sin Estudios	No	No	Si	Almacena el ultimo nivel académico del atleta
sabe_leer	Booleano	**True ó False**	1		False	No	No	Si	Almacena si el atleta puede leer.
sabe_escribir	Booleano	**True ó False**	1		False	No	No	Si	Almacena si el atleta puede escribir
sabe_firmar	Booleano	**True ó False**	1		False	No	No	Si	Almacena si el atleta puede firmar
edad_inicio	Integer	**99**	2	7-50	7	No	No	Si	Almacena la edad en la que el atleta empezó a surfear
idiomas	Carácter	**xxxx**	100	a-z A-Z	Español	No	No	Si	Almacena los idiomas que puede hablar el Atleta
otro_estudios	Carácter	**Xxxx**	500	a-z A-Z	No Posee	No	Si	Si	Almacena si el atleta tiene otros estudios
anios_practicando	Integer	**99**	2	0-40	0	No	No	Si	Almacena la cantidad de años que tiene el atleta de practicar SURF
rutina_constancia	Carácter	**xxxx**	500	a-z A-Z	No Posee	No	Si	Si	Almacena la descripción del Atleta
playa_practica	Carácter	**xxxx**	50	a-z A-Z	No Posee	No	Si	No	Almacena la playa donde practica el Atleta

lado_pie	Enum	**xxxx**	9	a-z A-Z	Derecha	No	No	Si	Almacena el lado del pie con que surfea el Atleta
ola_preferida	Enum	**xxxx**	50	a-z A-Z	No Posee	No	No	Si	Almacena la ola preferida por el Atleta
tiene_lesion	Boolean	**True ó False**	1		False	No	No	Si	Almacena si el Atleta ha sufrido alguna lesión , o impedimento físico, o de salud
descripcion_lesion	Carácter	**xxxx**	500	a-z A-Z	No Posee	No	Si	No	Almacena la descripción de la lesión , o impedimento físico, o de salud
compitio_fechas	Boolean	**True ó False**	1		True	No	No	Si	Almacena si el atleta a la hora del registro había participado en al fechas
cuantas_fechas	Entero	**99**	1	0-6	0	No	No	No	Almacena en cuantas fechas participo el surfista
ultima_participacion	Fecha	**99/99/9999**		0-9/	No Posee	No	Si	No	Almacena la fecha de última participación del Atleta
logros	Carácter	**xxxx**	200	a-z A-Z 0-9	No Posee	No	Si	Si	Almacena la descripción de logros del atleta
club_id	Entero	**99**	11	0-9	No posee	No	Si	No	Almacena el id de la entidad Club
escuela_id	Entero	**99**	11	0-9	No posee	No	Si	No	Almacena el id de la entidad Escuela

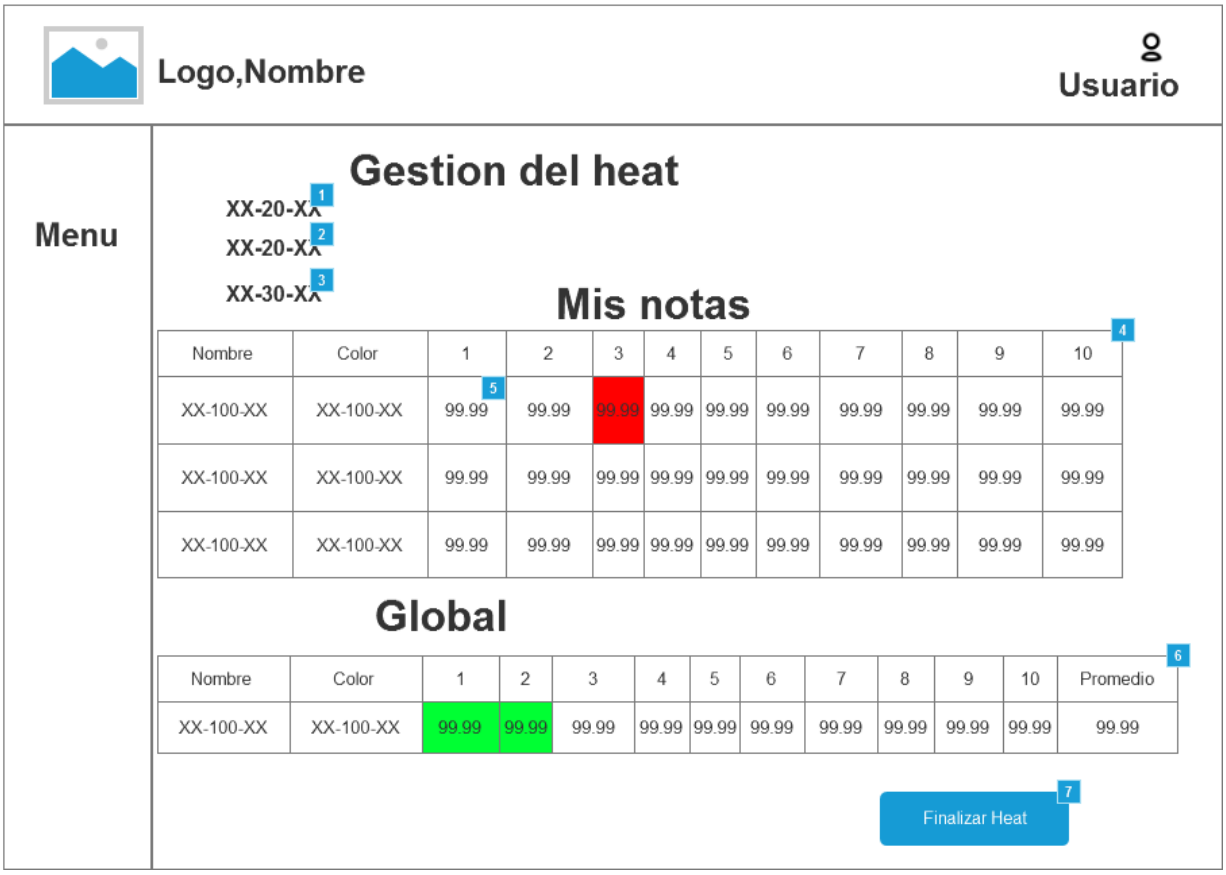
Relaciones foráneas: **Club, Escuela** Relaciones: **PatrocinadoAtleta, AtletaFecha, SurfistaPlaya, Ranking, Persona**	**Campos Clave:** ID, CLUB_ID, ESCUELA_ID

3.4. Diseño de interfaces

Gestión Heat

Pantalla que permitirá ver las puntuaciones de los atletas en el Heat tanto en detalle por cada juez como en resumen del Heat Global y definirá los atletas que pasaran a la siguiente Ronda.

Interfaz de usuario



Elementos

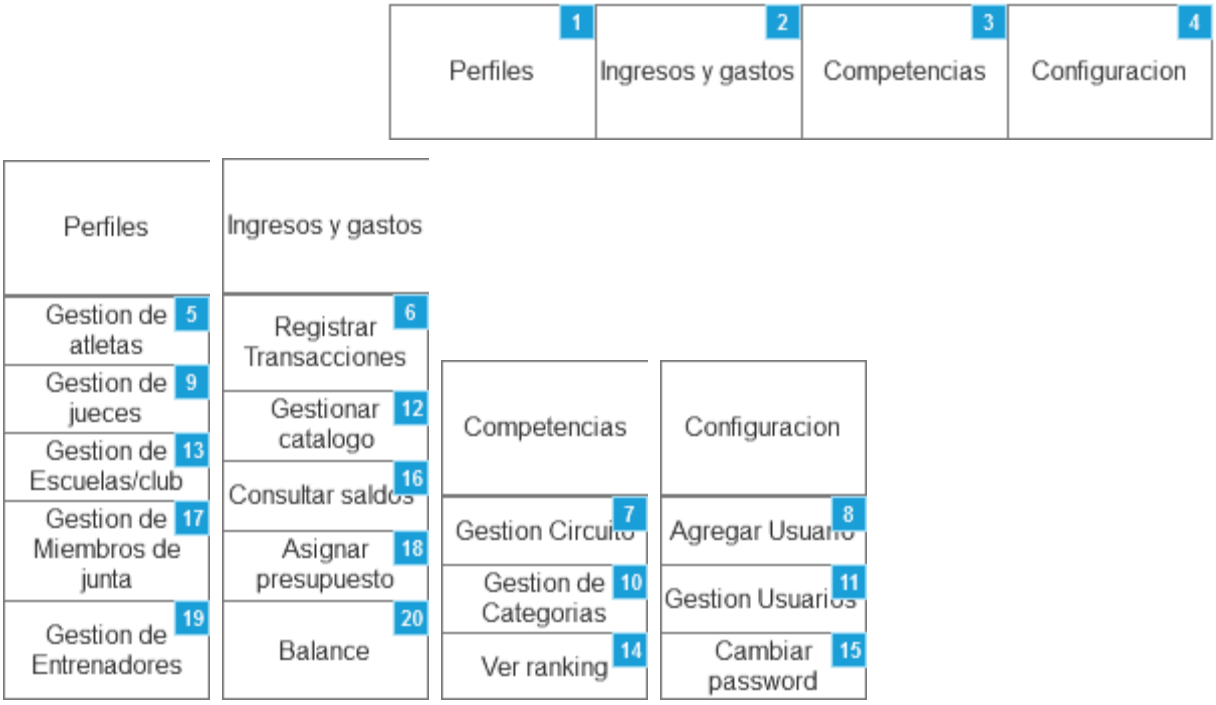
Numero	Nombre	Digitado	Seleccionado	Recuperado	Requerido	Formato
1	Nombre de la competencia			X		Alfanumérico
2	Ronda				X	Alfanumérico
3	Heat			X		Alfanumérico
4	Tabla de notas	X			X	99.99
6	Tabla global			X		99.99

Enlaces

Numero	Nombre	Enlace
5	puntuar	Enlace a Agregar puntuación en ventana emergente
7	Btn finalizar heat	Enlace a Resultados

Menú Principal

Organización de los menús



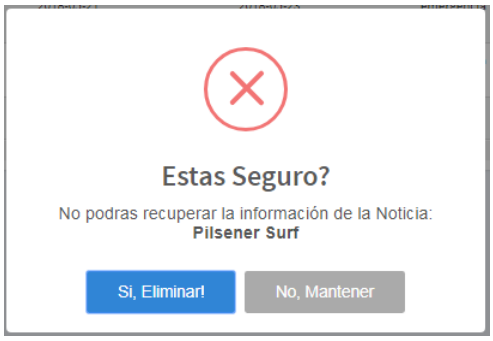
Enlaces

Numero	Nombre	Enlace	Acción
1	Menú superior de perfiles		Menú de perfiles
2	Gestión de ingresos y gastos		Menús de ingresos y gastos
3	Menú de competencias		Menú de competencias
4	Menú de Configuraciones		Menú de configuración
5	Gestión de atletas	Enlace a Gestión de atletas	
6	Registrar transacciones	Enlace a Registrar Transacción	
7	Gestión de circuitos	Enlace a Gestión de circuitos	
8	Agregar usuario	Enlace a Crear Usuario	
9	Gestión de jueces	Enlace a Gestión de jueces	
10	Gestión de categorías	Enlace a Gestión de Categorías	
11	Gestión de usuarios	Enlace a Gestión de usuarios	
12	Gestionar catalogo	Enlace a Gestionar Catalogo	
13	Gestión de Escuelas/Club	Enlace a Gestión de club/escuela	
14	Ver ranking	Enlace a Menú ranking	
15	Cambiar Password	Enlace a Actualizar contraseña	
16	Consultar saldos	Enlace a Consultar Saldos.	
17	Gestionar miembros de junta	Enlace a Gestión de Miembros de junta	
18	Asignar presupuesto	Enlace a Asignar Presupuesto	
19	Gestión de entrenadores	Enlace a Gestión de Entrenadores	
20	Balance	Enlace a Balance	

Mensajes de la Aplicación

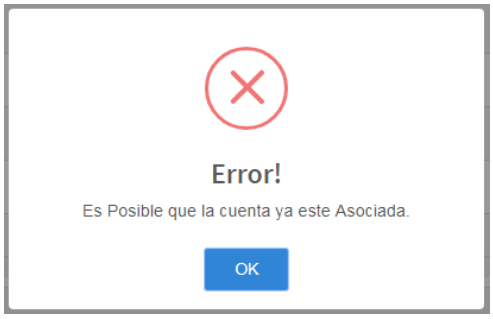
Mensaje de Confirmación:

Se utilizarán cuando se realice para todos los procesos que implique un cambio a nivel de base de datos



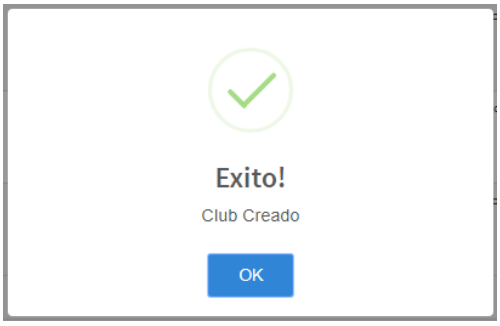
Mensaje de Error.

Mensaje de error se desplegará cuando cualquier acción en su flujo normal genere una excepción.



Mensaje de Éxito

Aparecerá al finalizar cualquier flujo normal.



Interfaz de usuario (App Móvil)

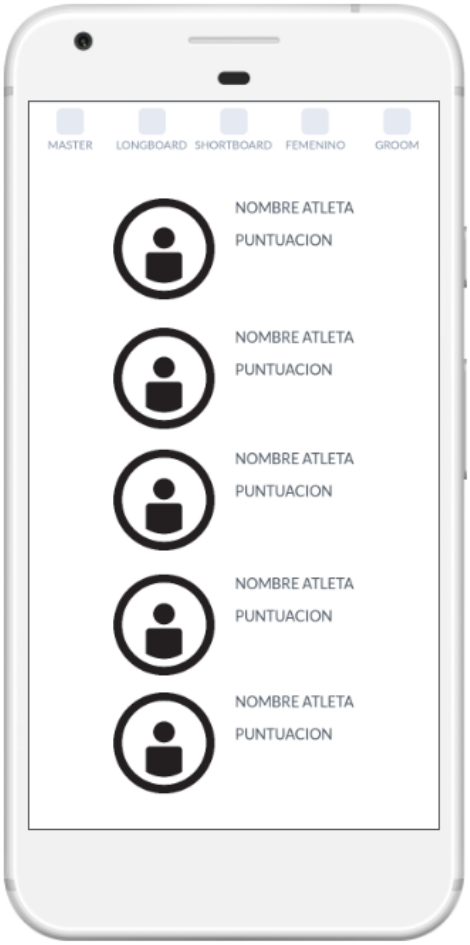
El Diseño de la App móvil que se utilizara para consulta de información responde al siguiente diseño.

Inicio

Menu

Competencias

Ranking



3.5. Diseño de Seguridad del Sistema

La seguridad para el sistema implementado en FESASURF tiene como objetivo primordial asegurar la confidencialidad, integridad y disponibilidad de la información para los usuarios autorizados, para esto se detalla una lista de los tipos de usuario que tienen acceso al sistema.

3.5.1. Usuarios del Sistema

Dentro de FESASURF las personas que tendrán acceso al sistema están alineadas en perfiles de acuerdo a las funciones que estos realizan en la entidad, estos usuarios son:

Tipo Usuario	Descripción del Perfil
Usuario Estándar	Es el usuario que solo necesita realizar consultas básicas acerca de información abierta al publico
Usuario Administrador	Este usuario es aquel que tiene acceso total al sistema
Juez Evaluador	Es el usuario encargado de gestionar las puntuaciones de los Heats que le competen

3.5.2. Roles

Tipo Usuario	Descripción
Usuario Estándar	Consultar Ranking, Ver Noticias, Buscar Competencia, Ver competencias
Usuario Administrador	Gestión y Control Total de las Actividades de FESASURF
Juez Evaluador	Gestionar de Puntuaciones

Conclusiones

Al finalizar el “Sistema Informático para la Gestión y Control de las Actividades de la Federación Salvadoreña de Surf (SWELL)”, se concluye lo siguiente:

- Con el análisis de la situación actual en FESASURF, se logró la comprensión de los procesos que se realizan, permitiendo identificar oportunidades de mejora., surgiendo la necesidad de un Sistema Informático que permita ayudar al personal en la gestión de la información.

- Se logró obtener los requerimientos de usuarios, los cuales darán solución a las oportunidades de mejoras encontradas. Para la determinación de requerimientos se tomó como base el análisis de la situación actual, para ello se hiso uso de herramientas de recolección de datos como lo son: la entrevista.

- Se elaboró el documento de Análisis y Diseño del Sistema, el cual sirvió como base para el desarrollo del mismo. Dicha documentación contiene la especificación de los estándares necesarios para la documentación, programación, base de datos e interfaces; los modelos de datos que reflejan la lógica del negocio, tales como modelo lógico, modelo físico de la base de datos, elaborando también el modelo del dominio y el diagrama de clases. Además, se diseñó las ventanas de entrada, salida e interfaces del Sistema.

- Con la implementación de SWELL se permitirá mejorar los procesos actuales en la federación, mejorando los tiempos en las actividades que se realizan, ya que SWELL permite una mejor gestión y control de las actividades de FESASURF.

- Se realizaron las pruebas necesarias para comprobar el correcto funcionamiento del Sistema, lo cual permitió encontrar errores y la solución de estos.

- Se elaboró la documentación necesaria para el Sistema Informático, siendo esta el manual de usuario, manual técnico y manual de instalación del software. Esta documentación será de gran ayuda para los usuarios del Sistema, permitiendo su fácil uso, además permitirá realizar mejoras futuras al Sistema.

- Se creó un plan de implementación para apoyar en la puesta en marcha del Sistema.

- Con la finalización de este proyecto, se beneficiará grandemente a la Federación Salvadoreña de Surf, permitiendo mejoras significativas en sus procesos actuales relacionados a la Gestión de competencias, Gestión de Atletas, Gestión de Entrenadores, Gestión de Clubes, Gestión de Fechas, Gestión de Rondas, Gestión de Heats, Gestión de Puntuaciones, Gestión de Ingresos y Gastos, entre otros.

Bibliografía

Libros:

- IZAMORAR. (2017). Beneficios, Importancia y Objetivos de un Sistema de Información. Recuperado de https://izamorar.com/beneficios-importancia-y-objetivos-de-un-sistema-de-informacion/
- Análisis y diseño orientado a objetos de sistemas usando UML. Autores: Bennett Simon, Ray Farmer, Steve McRobb.
- Análisis y diseño de sistemas /por Kenneth E. Kendall, Julie E. Kendall

Sitios Web Consultados:

- **Metodologías de Desarrollo de Software:** http://okhosting.com/blog/metodologias-del-desarrollo-de-software/
- **Ventajas y Desventajas de las Metodologías de Desarrollo de Software:** https://sites.google.com/site/metdlgsddesarrollodesoftware/modelo-incremental/cdesventajasdelmodeloincremental
- **Metodologías de Desarrollo de Software (RUP):** https://es.slideshare.net/cortesalvarez/metodologa-rup
- **Análisis de Requisitos de Software:** http://yaqui.mxl.uabc.mx/~molguin/as/IngReq.htm
- **Desarrollo de Casos de usos:** https://users.dcc.uchile.cl/~psalinas/uml/casosuso.html
- **Construcción de Diagramas de secuencia :** https://sg.com.mx/revista/45/reconociendo-los-diagramas-buen-comportamiento-diagramas-secuencia#.WaGfTz7ygdU
- **Diseño de estándares de programación:** https://sites.google.com/site/edmundoogaz/buenas-practicas-java
- **Documentación Oficial Framework Vue JS:** https://vuejs.org/
- **ECMAScript especificación del lenguaje que es basada en JavaScript:** https://www.w3schools.com/js/js_versions.asp
- **Diccionarios de datos para sistemas de información:** http://katyygaby.blogspot.com/p/diccionario-de-datos.html
- **Sitio Web Instituto Nacional de los Deportes de El salvador (INDES):** http://indes.gob.sv/
- **Sitio Web Comité Olímpico de el Salvador (COES):** http://www.teamesa.org/
- **Plan Gobierno Abierto de el Salvador:** http://www.gobiernoabierto.gob.sv/
- **Estatutos de la federación salvadoreña de surf (FESASURF)**: http://www.fesasurf.org.sv/archivos/estatutos_fesasurf.pdf
- **Estatutos de comité olímpico de el salvador (COES).** http://atletismoelsalvador.org/wp-content/uploads/2016/05/ESTATUTOS-COES-2004.pdf

Glosario de Términos

Swell:

Aplicación para la gestión de las actividades de la federación de surf.

Heat:

Grupos a competir durante la competencia, los Heat estarán compuestos, dependiendo la cantidad de atletas registrados para cada categoría, tomando en consideración que cada Heat debe de tener como mínimo tres atletas y máximo 4, aunque existe el caso excepcional de tener un máximo de 5 atletas en un Heat, que serán diferenciados por el color de sus licras clasificándose para la siguiente ronda los dos que obtengan una mejor puntuación.

En el desarrollo del Heat se selecciona el juez líder que rota en cada Heat entre los jueces participantes del circuito y las cabezas de Heat, es decir los atletas que tengan mejor rango no compiten entre ellos en las primeras rondas y se reasignan entre los heats.

Ronda:

Se realizarán n rondas con n Heat hasta que solo queden 4 atletas que serán los finalistas, las rondas como se menciona están compuestos de los heats, puntuaciones así como también atletas y jueces participantes en cada una de ellos que deben ser agregados posterior a la creación de un circuito.

Circuito:

Competencias que se realizan durante el año en diferentes categorías.

Arquitectura del sistema:

La arquitectura de software es el diseño de más alto nivel de la estructura de un sistema.

Fesasurf:

Federación salvadoreña de surf

Frontend:

Son todas aquellas tecnologías que corren del lado del cliente, es decir, todas aquellas tecnologías que corren del lado del navegador web.

Backend:

El programador Backend es aquel que se encuentra del lado del servidor, es decir, esta persona se encarga de lenguajes como PHP, Python, .Net, Java, etc.,

Entidad:

En bases de datos, una entidad es la representación de un objeto o concepto del mundo real que se describe en una base de datos.

Anexos

Formulario1

Federación Salvadoreña de Surf

FESASURF

Datos Generales

Nombre: _____ Apellidos: _____
Edad: _____ Sexo: _____ Estado: _____
Dui: _____ Nit: _____
Fecha de nacimiento: _____
Direccion: _____
Teléfono: _____ Correo electrónico _____
Categoría: _____ Playa local: _____

Datos académicos

Nivel Académico: _____ Último año cursado: _____
Sabe leer o escribir: si ( ) no ( ) Sabe firmar: si ( ) no( )
Cuantos idiomas sabe hablar (especifique) : _____
Otros estudios: _____

Datos Técnicos

Edad de inicio de surf: _____ cuantos años practicando: _____
Rutina y constancia de entreno (especifique) : _____
Lado del pie con que surfea: _____
Playa donde practica: _____
Tipo de ola preferida: _____
Ha sufrido alguna lesión o impedimento físico o de salud si ( ) no( )
Especifique: _____

atrocinadores y logros obtenidos

arca Patrocinadora: _____
empo de patrocinio: _____
ompitió todas las fechas del circuito? si ( ) no( ) ¿Cuántas?: _____
uál fue la última fecha de participación? _____
gros competitivos alcanzados: _____

FEDERACION SALVADOREÑA DE SURF
4ta Calle Poniente y 3era Ave. Sur, Playa La Paz, Edificio 199, La Libertad
Contiguo a Rest. Nuevo Altamar