UNIVERSIDAD TECNOLÓGICA NACIONAL

FACULTAD REGIONAL TUCUMÁN

DEPARTAMENTO DE INGENIERÍA EN SISTEMAS DE INFORMACIÓN

CÁTEDRA DE PROYECTO

Título del Proyecto:

"SOCIALDROID"

Autores:

Cristian Roberto Mene – Legajo 23719

Juan Ernesto Villegas – Legajo 20630

Tutor del Proyecto: Ing. Alberto Moyano

Año: 2014

Universidad Tecnológica Nacional
UTN – FRT
Cátedra de Proyecto

Quien suscribe, Ing. Alberto Moyano, acredita por medio de la presente que los datos entregados por el sistema “SocialDroid” satisfacen las necesidades del Usuario, cumpliendo el proyecto con los objetivos fijados y son aptos para su uso. Doy fe del trabajo realizado bajo mi tutoría.

……
Ing. Alberto Moyano
Tutor

A nuestras madres Carmen y Mirta,

nuestras familias,

nuestro tutor Ing. Alberto Moyano,

nuestros compañeros y amigos,

los que ya no están con nosotros,

los Profesores que nos marcaron el camino

Índice

Resúmen:

La exposición que se realiza en las siguientes páginas es un resumen de todas las actividades efectuadas para cumplir con el cronograma general de actividades establecido para la realización del Proyecto de Graduación.

El objetivo del proyecto es desarrollar un paquete de software a medida llamado “SocialDroid” para gestionar la integración de las distintas redes sociales en un sitio web en común, permitiendo a los usuarios mediante el registro de eventos publicar, leer y evaluar contenidos en la aplicación mediante los perfiles de cuenta de la red social inicial. Además de la difusión de contenidos e ideas propias de las personas conectadas hacia sus muros de Facebook (ó Twitter como se implementará a futuro), para el acceso de sus distintos seguidores internos y externos. En caso haber ocurrido un siniestro, los espectadores pueden facilitar el trabajo de las entidades responsables, indicando ubicación en Google Maps, horario, fecha, fotos y comentarios del mismo, utilizando la unidad de GPS del dispositivo del que cuente el navegante. Brindando al usuario final la posibilidad de utilizar sus propias cuentas de Facebook (ó Twitter como se implementará a futuro) para interactuar en nuestro portal, actualizando su estado en cada una de ellas y permitiendo que otros usuarios conectados a ellos mismos puedan visualizar la información. Estas publicaciones podrán ser compartidas a las redes sociales nativas.

El trabajo consistió en la determinación de la Metodología de Análisis y Diseño de Sistemas más conveniente que se debe aplicar en la situación problemática planteada, debiéndose optar entre la Metodología UML y el Diseño de Bases de Datos mediante el Modelo Entidad Relación y el Modelo Relacional con sus requerimientos, para lo cual se realizaron tareas de Análisis de Sistemas y Diseño de Bases de Datos como así también el Análisis y Diseño Orientado a Objetos y posteriormente la codificación o programación y prueba de la aplicación de software.

Algunas de las características más importantes de este software son las siguientes:

- ✓ Permite administrar un elevado número de usuarios, es decir es multiusuario.
- ✓ Permite registrar todos los eventos que ingresen los usuarios a través de sus respectivas cuentas de redes sociales.
- ✓ Permite registrar en Google Maps la ubicación donde ocurrió un siniestro.
- ✓ Permite que las entidades responsables de actuar en caso de siniestro puedan enterarse y actuar de manera rápida ante la ocurrencia del mismo.

Tema: SocialDroid

Ficha Técnica:

- Plataforma: Android, Web.
- Lenguaje: Java
- Dispositivos: Smartphones, Pc.
- Tipo: Apk sobre Redes Sociales y Comunicación.

Descripción:

SocialDroid interactuará con distintas redes sociales para permitir a los usuarios de ellas, publicar, leer y evaluar contenidos en la aplicación mediante los perfiles de cuenta de la red social inicial (Ej. Facebook).

Las publicaciones contarán con un sistema de puntaje a las mismas y a sus usuarios conformando distintos rankings.

Los eventos de distinta índole expuestos podrán ser ubicados en Google Maps mediante la unidad de GPS del dispositivo del que cuente el navegante.

Las publicaciones podrán ser compartidas a las redes sociales nativas.

Se utilizará un sistema RSS mediante Google Reader para que las personas puedan suscribirse a los distintos contenidos.

Objetivos perseguidos

Objetivos Teóricos

Los principales objetivos teóricos del proyecto son los siguientes:

- Analizar los antecedentes de la aparición de nuevas redes sociales y herramientas integradoras de las mismas en Internet.
- Identificar y evaluar los recursos informáticos y culturales para la realización de la herramienta de integración de redes sociales.
- Investigar las distintas alternativas de herramientas, aplicaciones y webs de redes sociales.

Objetivos Prácticos

- Elaborar un sistema informático que realice las siguientes tareas:
  - ✓ Permitir la registración mediante plugins de redes sociales en la herramienta.
  - ✓ Permitir la publicación de diversos contenidos compartidos entre las redes sociales conectadas a la herramienta.
  - ✓ Almacenar en una base de datos los datos proporcionados por los usuarios así como las estadísticas y resultados obtenidos luego del procesamiento de dichos datos.

- ✓ Emitir informes sobre el acceso a la herramienta para facilitar la venta del producto a los distintos posibles anunciantes de publicidad.

- Lograr alta calidad en el sistema informático. El sistema debe cumplir con los siguientes factores internos y externos de calidad:
  - ✓ Corrección: Se busca que el software desarrollado sea capaz de realizar con exactitud su propósito, tal y como definen las especificaciones.
  - ✓ Robustez: El software debe ser capaz de reaccionar apropiadamente ante situaciones excepcionales. Este factor complementa la corrección. Siempre habrán casos que no estén explícitamente contemplados en las especificaciones, lo que se busca es que si tal caso surgiese, el sistema no causará eventos catastróficos, arrojando mensajes de error convenientes y terminando su ejecución limpiamente en lo posible.
  - ✓ Extensibilidad: Para lograr la extensibilidad se tratará de, en las mejoras, seguir una simplicidad del diseño y una descentralización que proporcione mayor autonomía a los módulos desarrollados.
  - ✓ Reutilización: En lo posible se intentarán encontrar patrones comunes de manera de explotar esta similitud y evitar reinventar soluciones a problemas que fueron tratados con anterioridad.
  - ✓ Eficiencia: Se busca lograr la capacidad del software resultante de exigir la menor cantidad de recursos hardware.
  - ✓ Facilidad de uso: Se buscará que ante personas con distintas formaciones y aptitudes sean capaces de aprender a usar el sistema. Este factor también cubrirá las facilidades de instalación, de operación y de supervisión.
  - ✓ Funcionalidad: Se tratará de lograr un equilibrio entre la pérdida de consistencia, la facilidad de uso y la complejidad del sistema.

Estado de arte del tema

Justificación

Nuestro proyecto se basa en el conocimiento de la proliferación en la comunidad de nuevas tecnologías que permiten la comunicación eficaz de ideas y contenidos.

Con el auge de los teléfonos inteligentes, aparecieron los primeros sistemas operativos para móviles, como por ejemplo el SO propietario Symbian o Windows Phone. Pero debido a su condición de software enlatado y limitado por sus propietarios los desarrollos para los mismos no fueron abundantes. Con el surgimiento del SO libre de Google “Android”, los profesionales y desarrolladores del área vieron la oportunidad de

realizar software de mayor alcance al público gracias a un abanico de posibilidades brindado por el sistema mucho mayor a los anteriores. Hoy en día se pueden encontrar muchos tipos de aplicaciones para las necesidades del público en general pero aún puede detectarse que la diversidad de contenidos no se encuentra encapsulada en un único sitio o aplicación.

Este inconveniente podrá ser solucionado mediante nuestra herramienta de integración de redes sociales “SocialDroid”.

Mediante el registro de eventos por parte de los usuarios del sistema permitirá una comunicación eficaz e instantánea de noticias de interés público y sucesos de carácter político, económico, cultural y de cualquier otra índole. También permitirá la difusión de contenidos e ideas propias de las personas conectadas hacia sus muros, dando la posibilidad de utilizar sus propias cuentas de Facebook (ó Twitter como se implementará a futuro) para el acceso de sus distintos seguidores internos y externos.

A través del uso de Smartphones los integrantes de la red social podrán publicar fotos y comentarios sobre los eventos transmitidos. En caso de siniestros información de los espectadores pueden facilitar el trabajo de las entidades responsables, indicando ubicación en Google Maps, horario, fecha, fotos y comentarios del mismo.

Un sistema de ranking de publicaciones y usuarios ayudará a brindar a los lectores un marco de seguridad y veracidad sobre los contenidos accedidos. Los mismos usuarios podrán contar con reputaciones individuales para poder difundir sus materiales.

La implementación de RSS (Sindicación Realmente Simple o RSS 2.0) nos permitirá brindarle los contenidos a los que cada usuario este suscripto de forma amena y sencilla. Dicho formato de tipo XML enumerará las novedades del sitio como títulos, fechas o descripciones de los artículos. RSS estará conectado al agregador Google Reader para que el lector encuentre la herramienta lo más intuitiva posible.

Beneficios

Cualitativos:

- Inserción en el mercado del software para móviles,
- Amplia difusión de la aplicación mediante Google Play y portales de Internet.
- Posibilidad de ingresar en un “océano azul” aun no explotado en este sector del país.

Cuantitativos:

- Ingreso de beneficios monetarios por publicidad,
- Posibilidad de atraer usuarios desde las grandes redes sociales,

- Facilidad para implementar y distribuir la aplicación,
- Ventajas inherentes al software libre.

Restricciones

- SocialDroid solo trabajara con el sistema operativo Android[1].
- El lenguaje de programación deberá ser necesariamente PHP[2].
- SocialDroid solo soportará las tecnologías populares en el mercado en los últimos tiempos.

Situación a Nivel Nacional

Con el auge de las redes sociales en el mundo, Argentina se situó dentro de Latino América como uno de los países donde sus habitantes pasan más horas dentro de los portales, con un promedio superior a las casi 10hs mensuales por visitante solo en la red Social Facebook.

Las redes sociales permitieron a muchas empresas en nuestro pais posicionarse de cara al mercado y acceder al público de forma más directa y rápida que los medios convencionales.

Entre las redes sociales que nacieron en nuestro país podemos nombrar a Sónico con más de 50 millones de miembros y oficinas tanto en Buenos Aires como San Pablo (Br) y Miami (EEUU).

En los últimos años las redes sociales están enfrentando el hecho que los usuarios pasan de un portal a otro buscando contenido sin regresar al primero por lo que comenzaron una carrera por la captación de la atención del usuario.

La oportunidad surge de poder concentrar en un solo portal esos contenidos que son brindados por las distintas redes sociales sin dificultar su acceso al mismo.

Situación a Nivel Internacional

Las grandes redes sociales en el mundo surgieron de las tres siguientes necesidades:

[1] Sistema operativo basado en Linux diseñado principalmente para dispositivos móviles con pantalla táctil, como teléfonos inteligentes o tabletas.

[2] Lenguaje de programación de uso general de código del lado del servidor originalmente diseñado para el desarrollo web de contenido dinámico.

1. Comunicación.
2. Integración.
3. Cooperación.

Comunicación: “Poder de expresión en manos de personas comunes” los medios de comunicación convenciones permiten solo la difusión de algunas opiniones o ideas, las personas internamente sienten la necesidad de ser escuchados o mostrarle al mundo sus logros o eventos importantes. Con el surgimiento de los primeros Blogs o páginas webs personales se fue allanando el camino a lo que las redes sociales representan hoy en día, la libertad de expresar ideas pero también poder escuchar a personas desconocidas o de lugares remotos que de otra forma seria impensado.

Integración: “Ser parte del todo”. Desde tiempos remotos la humanidad tuvo la necesidad de estar en comunidad, conocer diversas culturas y aprender. Todos estos puntos pueden obtenerse de las redes sociales hoy en día por la masificación de su uso en el globo.

Cooperación: “El trabajo en equipo”. Hoy en día muchas empresas lejos de bloquear el uso de este tipo de herramientas, las utilizan para su desenvolvimiento cotidiano en especial cuando la distancia hace más practico el uso de las mismas que el uso del teléfono o el correo electrónico convencional. Por ejemplo el uso de Skype, Dropbox, Twitter, LinkedIn, etc.

Desarrollo

Metodología de trabajo

Planificar y ejecutar un proyecto requiere de una metodología que guíe al equipo, para que sus esfuerzos sean aprovechados al máximo. Inicialmente se realizarán los estudios y capacitaciones planteados en los objetivos teóricos, a fin de obtener los conocimientos necesarios para una profunda comprensión de las tareas a desarrollar en los objetivos prácticos, y eficiencia a la hora de realizarlos.

Para resolver la situación problemática planteada, como punto de partida, se aplicó la Metodología de Diseño de Base de Datos mediante el Modelo Entidad Relación y el Modelo Relacional, realizándose para ello tareas tales como Análisis y Captura de Requerimientos, Relevamiento[3], Creación del Modelo de Datos del Usuario (Modelo Entidad-Relación), Construcción de Base de Datos Relacional y Diseño de la Aplicación[4].

3

4 **Bibliografía guía: Sistemas Administrativos – Magdalena, Fernando.**

Bibliografía guía: Procesamiento de Bases de Datos – Kroenke, David

Luego, al momento de realizar el sistema de información planteado en los objetivos prácticos se seguirá la siguiente metodología de trabajo:

1. Análisis Preliminar

En esta primera etapa se determinan las características que serán plasmadas en el software.

Considerando esto se elabora una propuesta en la cual se determina el alcance, y se estiman plazos y costos.

Si bien esta es una primera aproximación sirve de límite para las siguientes etapas.

1.1 Análisis y captura de requerimientos

Para obtener las especificaciones de los requerimientos de información necesarios para desarrollar la aplicación y construir el modelo de datos del usuario se realizaron las siguientes actividades específicas:

1.2 Relevamiento:

Consistió en la realización de las siguientes actividades:

- Recolección externa de datos.
- Observación directa.
- Organización de la información.

1.2.1 Recolección externa de datos:

Consistió en la búsqueda de información a través de **Internet** para **conocer** cómo funcionan en detalle las redes sociales de mayor auge en la actualidad, siendo éstas Facebook (ó Twitter como se implementará a futuro), observándose que no existen sistemas de software disponibles actualmente en el mercado capaces de combinarlas en un único sitio para maximizar potencialmente su facultad informativa, conocer las funciones que estos realizan, los módulos de software incorporados y el diseño de las pantallas de cada uno de ellos.

1.2.2 Observación directa:

Esta técnica de recolección de información permitió tomar conocimiento de los siguientes temas:

- ✓ La forma en que se realizan los procesos formales e informales de las distintas redes sociales.
- ✓ La forma en que se categorizan, califican, buscan, comparten y registran los eventos.
- ✓ La forma en que se registran y acceden los usuarios a las distintas redes sociales.
- ✓ La forma en que se utilizan los avisos publicitarios en las mismas para recaudar fondos.

Los resultados obtenidos se registraron en papel, realizándose posteriormente la organización en la etapa correspondiente.

Procesos observados:

- Logueo de usuarios
- Cerrar sesión de usuarios
- Compartir Eventos
- Categorizar Eventos
- Búsqueda de Eventos
- Registro de Eventos
- Calificar Eventos
- Ubicar eventos

Registración de procesos:

Para registrar los procesos observados más importantes se utilizó la **técnica** de la toma de notas.

1.2.3 Organización de la información recolectada:

Consistió en las siguientes actividades:

a) **Cruce de datos:** consistió en comparar los datos captados en la recolección de información externa realizada en Internet con los captados en la observación directa, **según el fluir de un determinado proceso**, y en caso de detectarse incoherencias se procedió a revisar detalladamente los procesos.

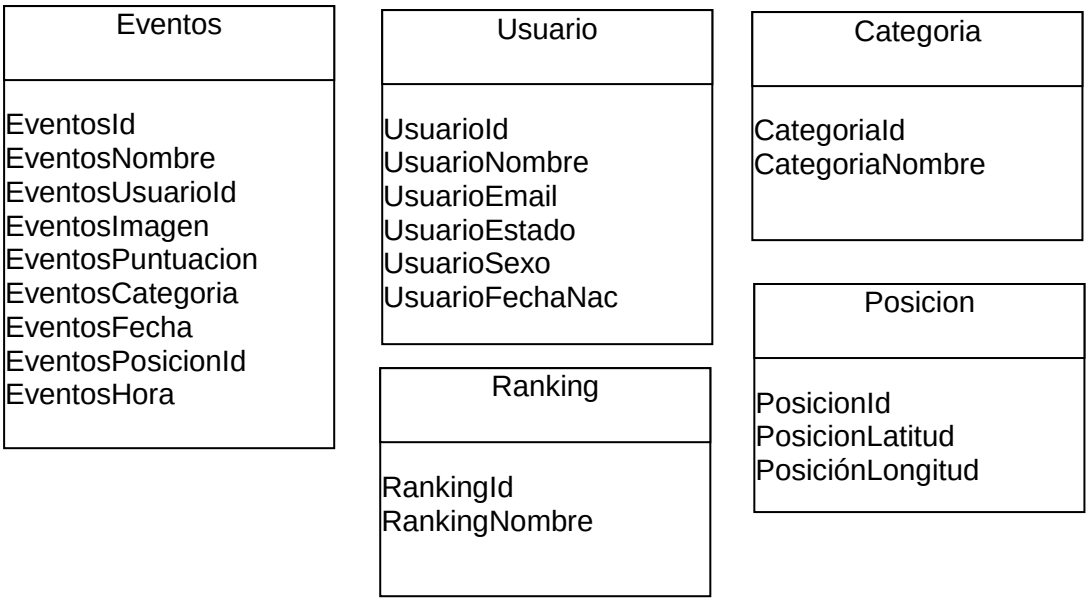
b) El **ordenamiento** de la información se realizó **en forma iterativa**, es decir que no se esperó a terminar el relevamiento de la totalidad de la información para luego ordenarla.

c) Se organizó una **carpeta** con la información obtenida.

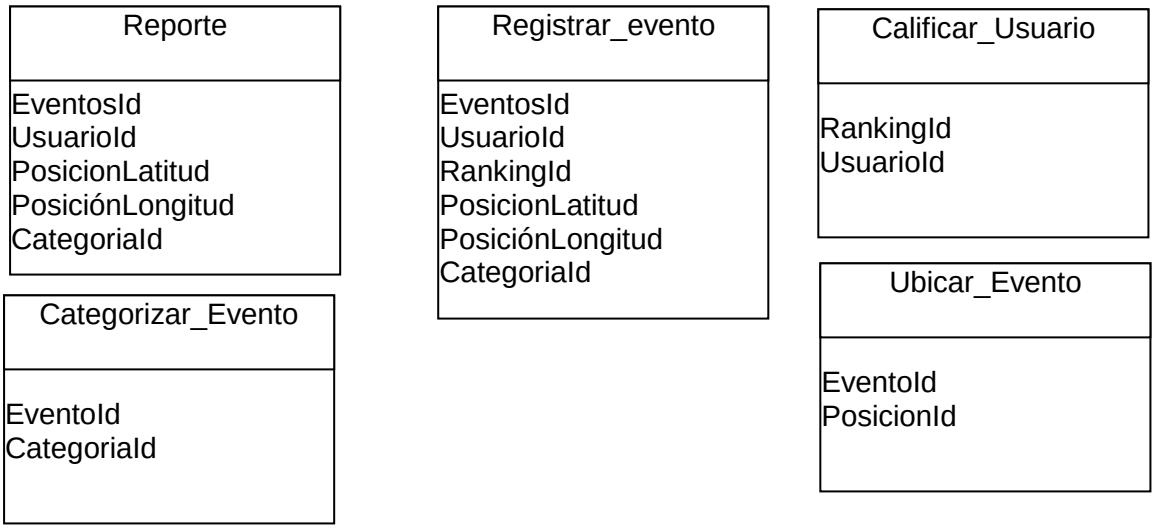
1.3 Creación del modelo de datos del usuario:

La meta más importante de la fase de requerimientos es la creación de un modelo de datos del usuario, el cual se implementó mediante el Modelo Entidad Relación de Peter Chen[5].

1.3.1 Las Entidades y sus Atributos:



1.3.2 Las Relaciones y sus Atributos:



5 Bibliografía guía: Procesamiento de Bases de Datos – Kroenke, David

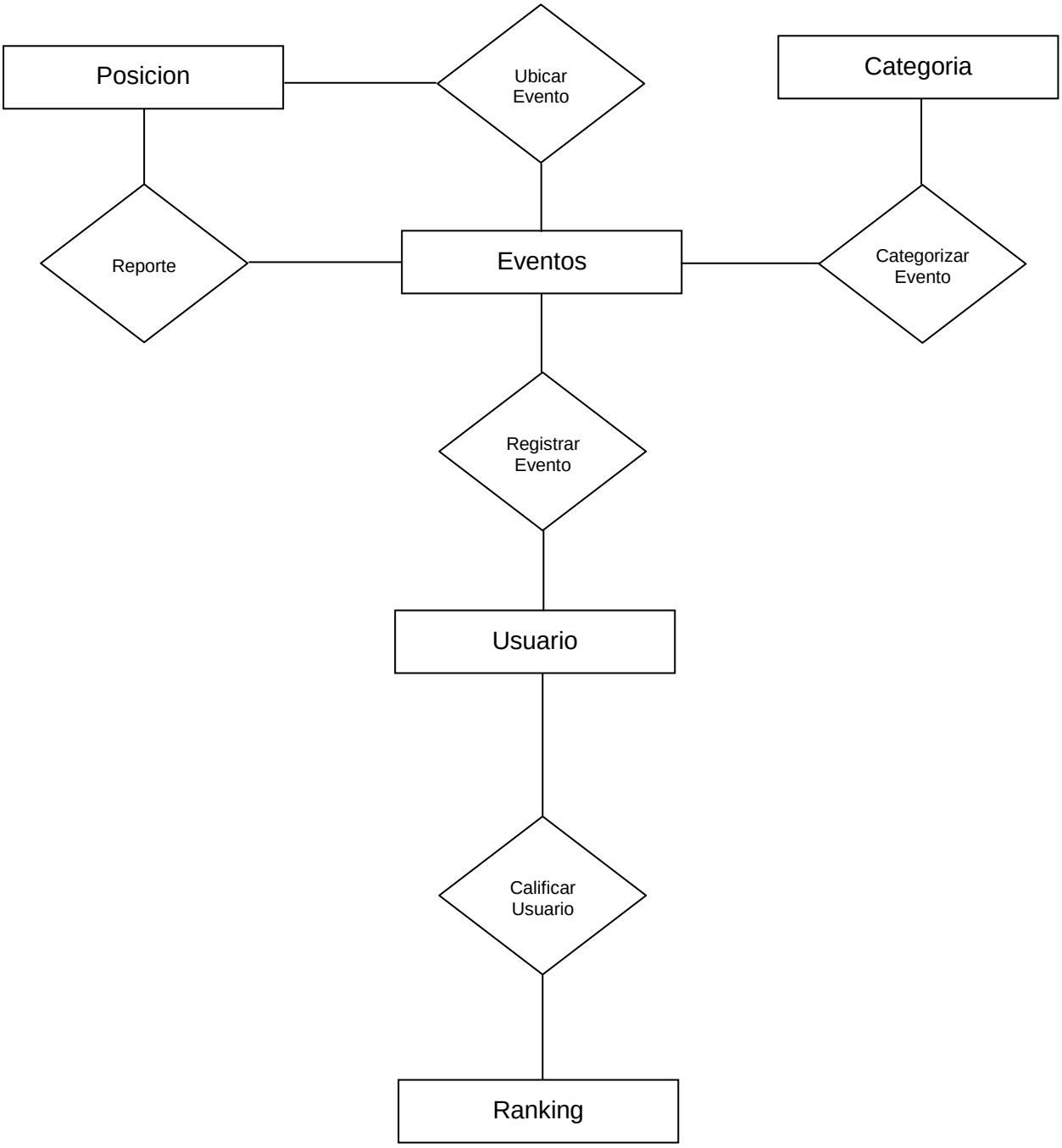
1.3.3 Las Reglas del Sistema y su documentación:

Las reglas del sistema son limitaciones en las actividades que deben reflejarse tanto en el modelo de datos del usuario, en la base de datos y en la aplicación.

Las reglas identificadas para nuestro sistema son las siguientes:

- Para que ingrese un usuario a nuestro sistema debe contar de forma obligatoria con una cuenta de Facebook (ó Twitter como se implementará a futuro), de lo contrario deberá proceder a la registración del mismo en alguna de estas redes sociales.
- El sistema contará a futuro con un moderador que no deberá permitir que se publiquen eventos con contenido ofensivo, ni perturbador a la sociedad.

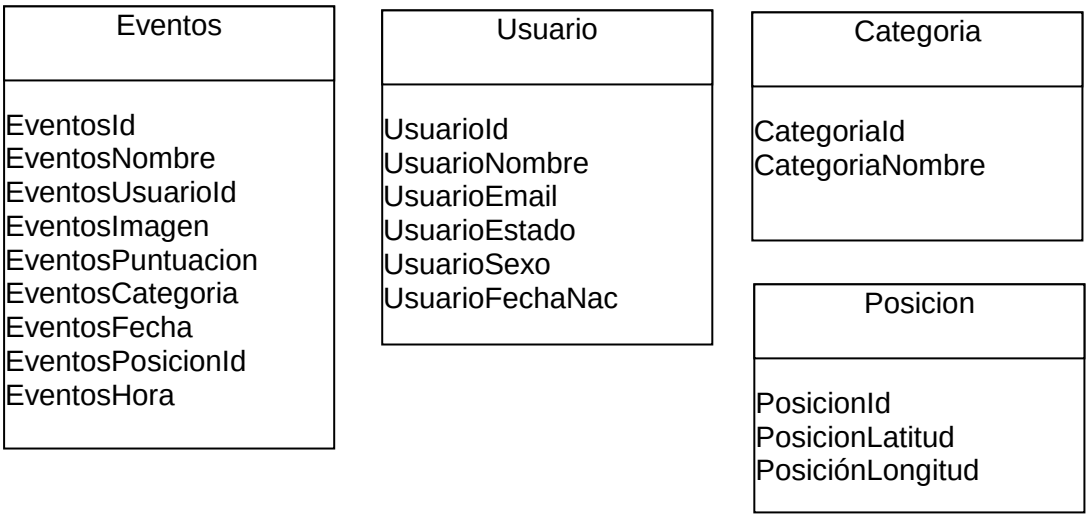
1.3.4 **Modelo Entidad Relación**



1.3.5 Reducción del Modelo Entidad Relación a un conjunto de tablas. Eliminación de tablas redundantes

Luego de aplicar la Metodología para reducir un Diagrama del Modelo Entidad Relación a un conjunto de tablas y de eliminar las tablas redundantes, se obtiene la base de datos buscada resultando el siguiente conjunto.

Tablas resultantes de la DB:



2. Determinación de las herramientas de software a usar:

Se realizo una evaluación de las distintas herramientas de software existentes en el mercado para desarrollar aplicaciones de bases de datos, para representar el Modelo Relacional y para gestionar Bases de Datos.

Las características excluyentes para su selección fueron:

Software para codificar la aplicación:

- Permitir el desarrollo de aplicaciones multiusuario.
- Poseer capacidad para trabajar con un motor de base de datos relacional.
- Poseer la tecnología de orientación a objetos.

Resultado: se prefirió **PHP, Adobe Dreamweaver CS6.**

Motor de Base de Datos:

- Potencia
- Soportar el Modelo Relacional
- Open source (libre)

Resultado: se optó por **MySQL 5**.

Servidor Web:

- Apache

Herramienta para modelar la Base de Datos:

- Soportar el Modelo Relacional
- Reflejar la Base de Datos modelada mediante el Modelo Relacional que posteriormente será implementada mediante el Motor de Base de Datos.

Resultado: se utilizó **ER/Studio v7.1.2** y **dbForge Studio for MySQL**.

3. Construcción de db relacional y diseño de la aplicación

Se definirá el proyecto con máximo nivel de detalle, se generan modelos del aspecto gráfico, del contenido, y del funcionamiento, abarcando todos los módulos a incluirse que resultaron de la fase de análisis.

Estos modelos son prototipos del proyecto y lo reflejan con exactitud, se trabajará evolucionándolos hasta que se quede satisfecho con todos los aspectos.

Es fundamental la aprobación de los mismos para seguir avanzando, pues los cambios en etapas posteriores serán más costosos.

3.1 Construcción de la Base de Datos Relacional:

3.1.1 Conversión del Modelo Entidad Relación al Modelo Relacional:

Pasos:

1. Se asigna y se define una Relación para cada tabla resultante del Modelo Entidad Relación.
2. El nombre de la Relación es el de la tabla del Modelo Entidad Relación.
3. Los atributos de la Relación son los de la tabla correspondiente del Modelo Entidad Relación.
4. Normalizar las Relaciones definidas, es decir aplicar a cada Relación las Formas Normales.
5. Modelar las Relaciones con una herramienta adecuada.

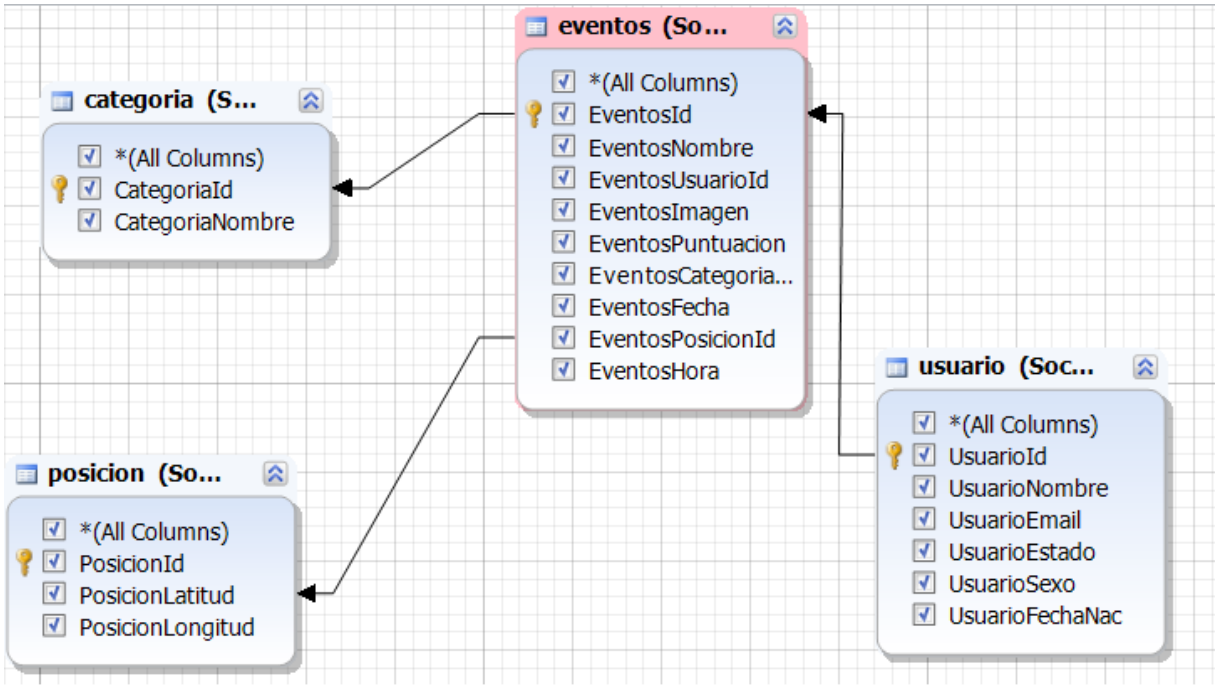
Para modelar el Modelo Relacional se utilizó la herramienta **dbForge Studio for MySQL**.

La secuencia de pasos es la siguiente:

1) Definir las tablas con sus respectivos campos y atributos
2) Generar un modelo Físico de la Base de Datos
3) Exportar el modelo Físico a Script

En la página siguiente se muestra el Modelo Relacional creado con **dbForge Studio for MySQL**

3.1.2 Modelo Físico del Modelo Relacional



Modelo Relacional creado con **dbForge Studio for MySQL**

3.1.3 Construcción de la Base de Datos física:

La Base de Datos en disco fue generada automáticamente por la herramienta ER/Studio a partir de los Modelos Lógico y Físico creados con anterioridad.

La secuencia de pasos fue la siguiente:

- Construir el Modelo Lógico de la base de datos
- Construir el Modelo Físico de la base de datos
- Seleccionar del menú principal del software ER/Studio la opción **Generar Base de Datos**.
- Definir el alias para conectarse con la base de datos
- Crear una nueva carpeta en el **directorio data** del motor de base de datos.
- Escritura en disco de las tablas de la base de datos.

A continuación se observa la estructura de las tablas de la base de datos creada con ER/Studio para nuestro sistema y una vista global de la base de datos.

Script que generará la BD SocialDroid.SQL

```
-- Base de datos: `socialdroid`
--
-- --------------------------------------------------------
--
-- Estructura de tabla para la tabla `categoria`
--
CREATE TABLE `categoria` (
  `CategoriaId` int(11) NOT NULL auto_increment,
  `CategoriaNombre` varchar(255) default NULL,
  PRIMARY KEY  (`CategoriaId`)
) ;

--
-- Volcar la base de datos para la tabla `categoria`
```

```
--
-- ------------------------------------------------------
--
-- Estructura de tabla para la tabla `eventos`
--
CREATE TABLE `eventos` (
  `EventosId` int(11) NOT NULL auto_increment,
  `EventosNombre` varchar(255) default NULL,
  `EventosUsuarioId` int(11) default NULL,
  `EventosImagen` blob,
  `EventosPuntuacion` varchar(255) default NULL,
  `EventosCategoriaId` int(11) default NULL,
  `EventosFecha` date default NULL,
  `EventosPosicionId` int(11) default NULL,
  `EventosHora` time default NULL,
  PRIMARY KEY  (`EventosId`)
);

--
-- Volcar la base de datos para la tabla `eventos`
--
-- ------------------------------------------------------
--
-- Estructura de tabla para la tabla `posicion`
--
CREATE TABLE `posicion` (
  `PosicionId` int(11) NOT NULL auto_increment,
  `PosicionLatitud` int(11) default NULL,
  `PosicionLongitud` int(11) default NULL,
  PRIMARY KEY  (`PosicionId`)
);

--
-- Volcar la base de datos para la tabla `posicion`
--
```

```
-- --------------------------------------------------------
--
-- Estructura de tabla para la tabla `usuario`
--
CREATE TABLE `usuario` (
  `UsuarioId` int(11) NOT NULL auto_increment,
  `UsuarioNombre` varchar(100) default NULL,
  `UsuarioEmail` varchar(100) default NULL,
  `UsuarioEstado` int(11) NOT NULL default '0',
  `UsuarioSexo` varchar(100) default NULL,
  `UsuarioFechaNac` date default NULL,
  PRIMARY KEY  (`UsuarioId`)
);

--
-- Volcar la base de datos para la tabla `usuario`
--
```

3.1.4 Estructura de las Tablas de la Base de Datos

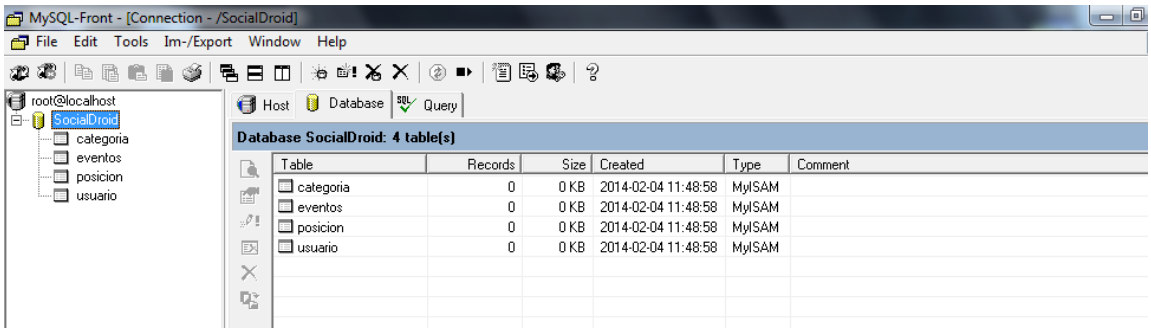


Tabla Categoría

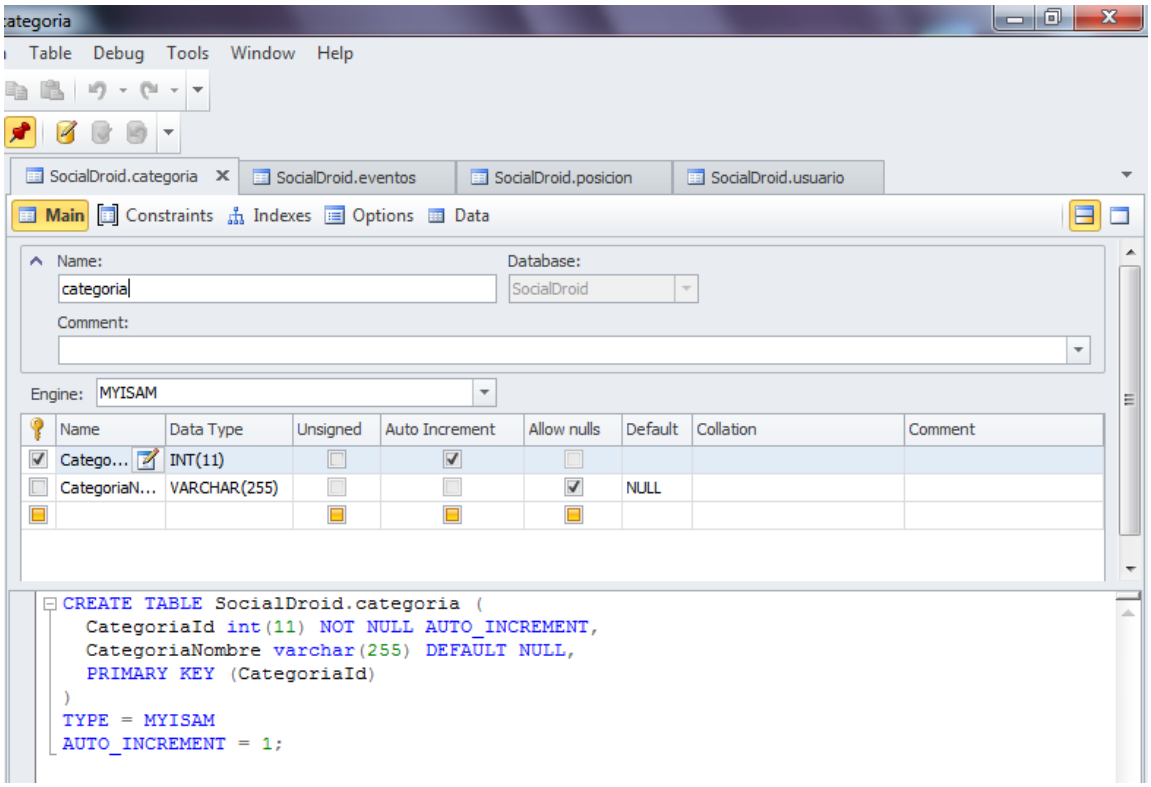


Tabla Eventos

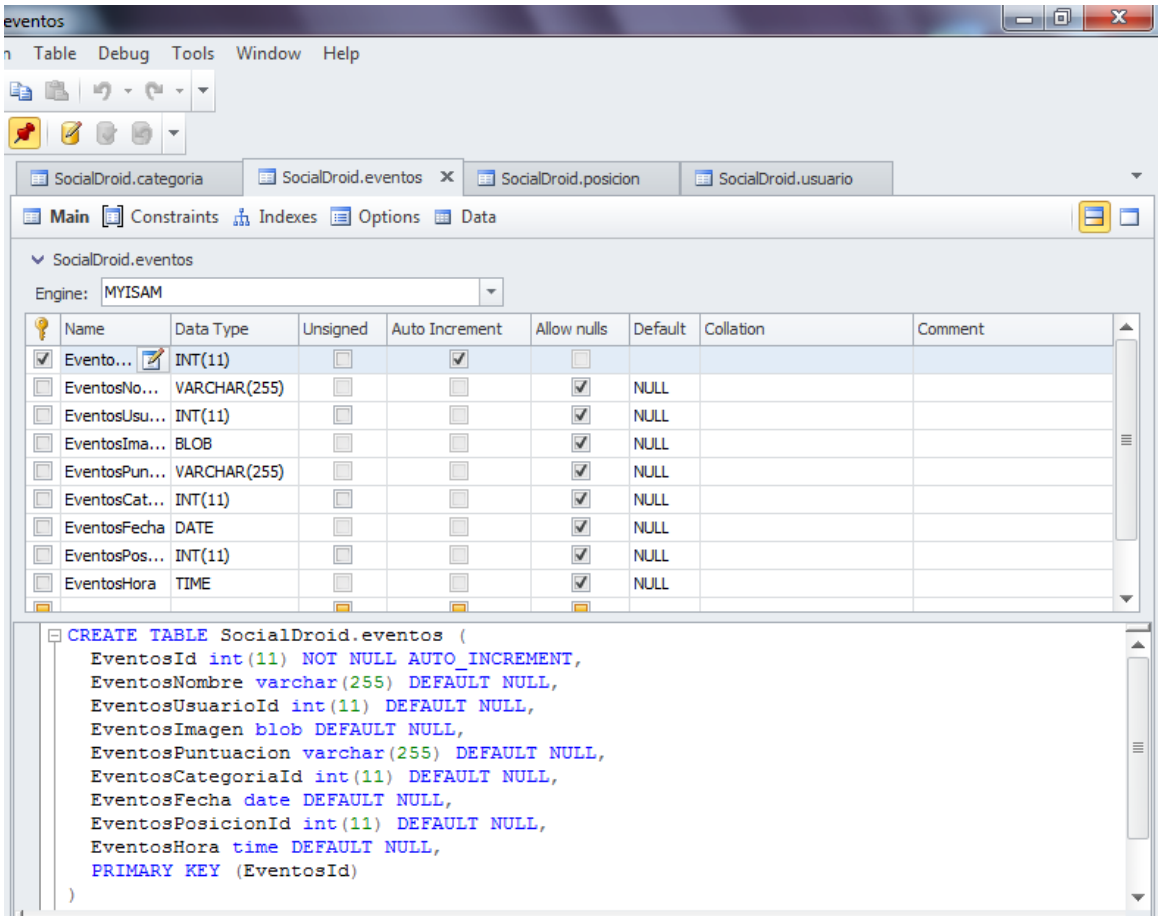


Tabla Posición

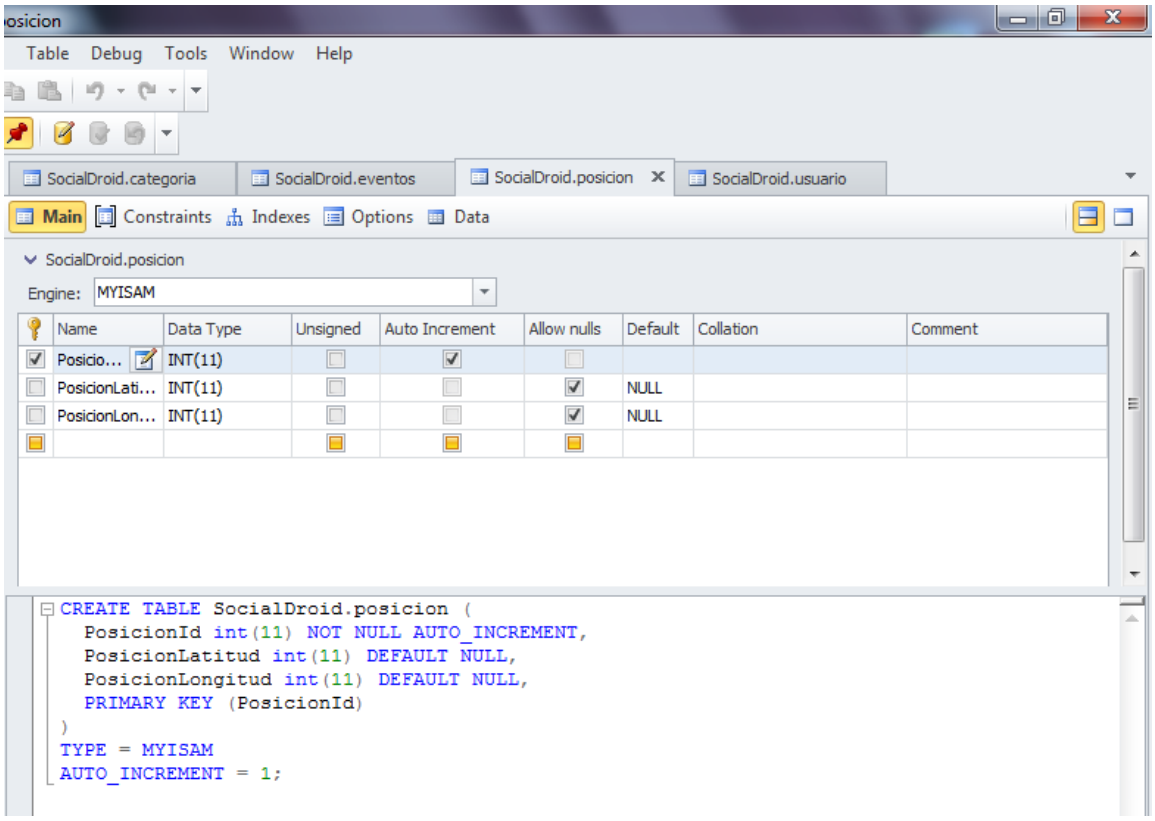
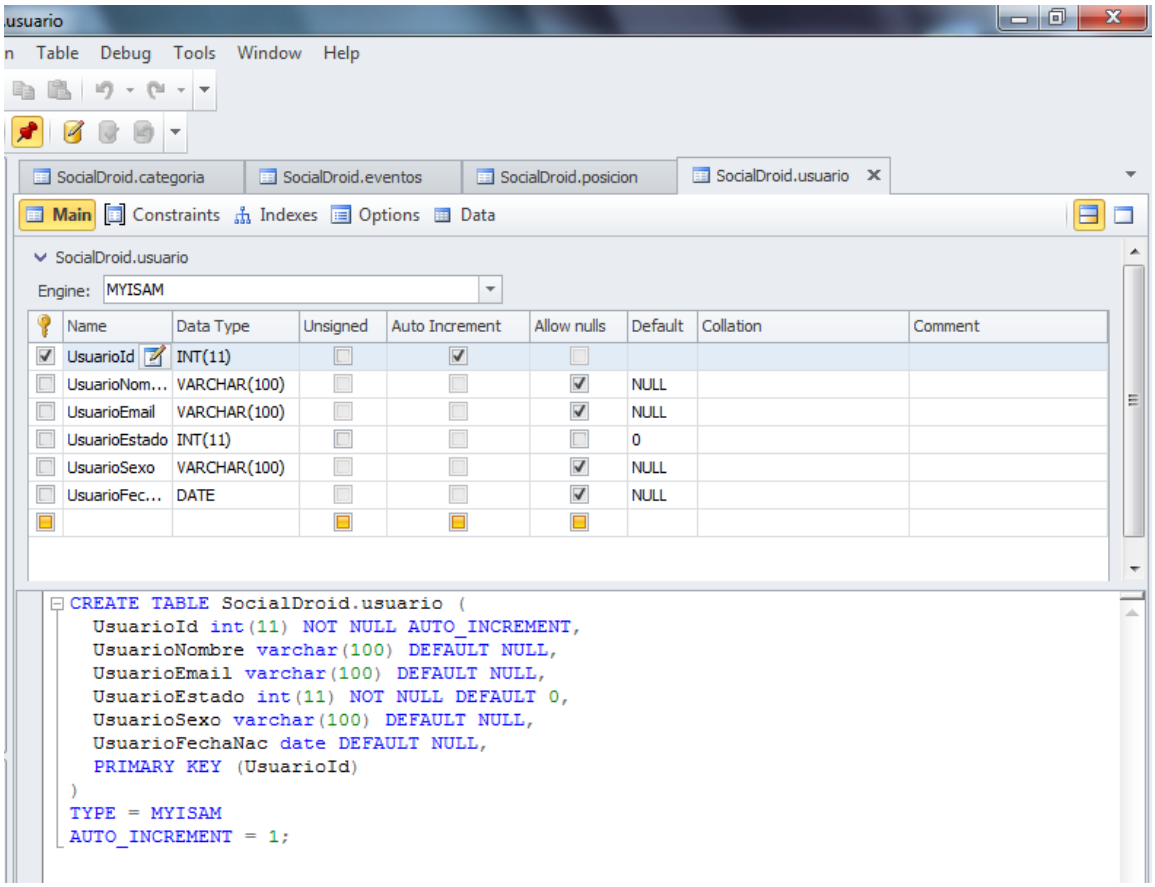


Tabla Usuario



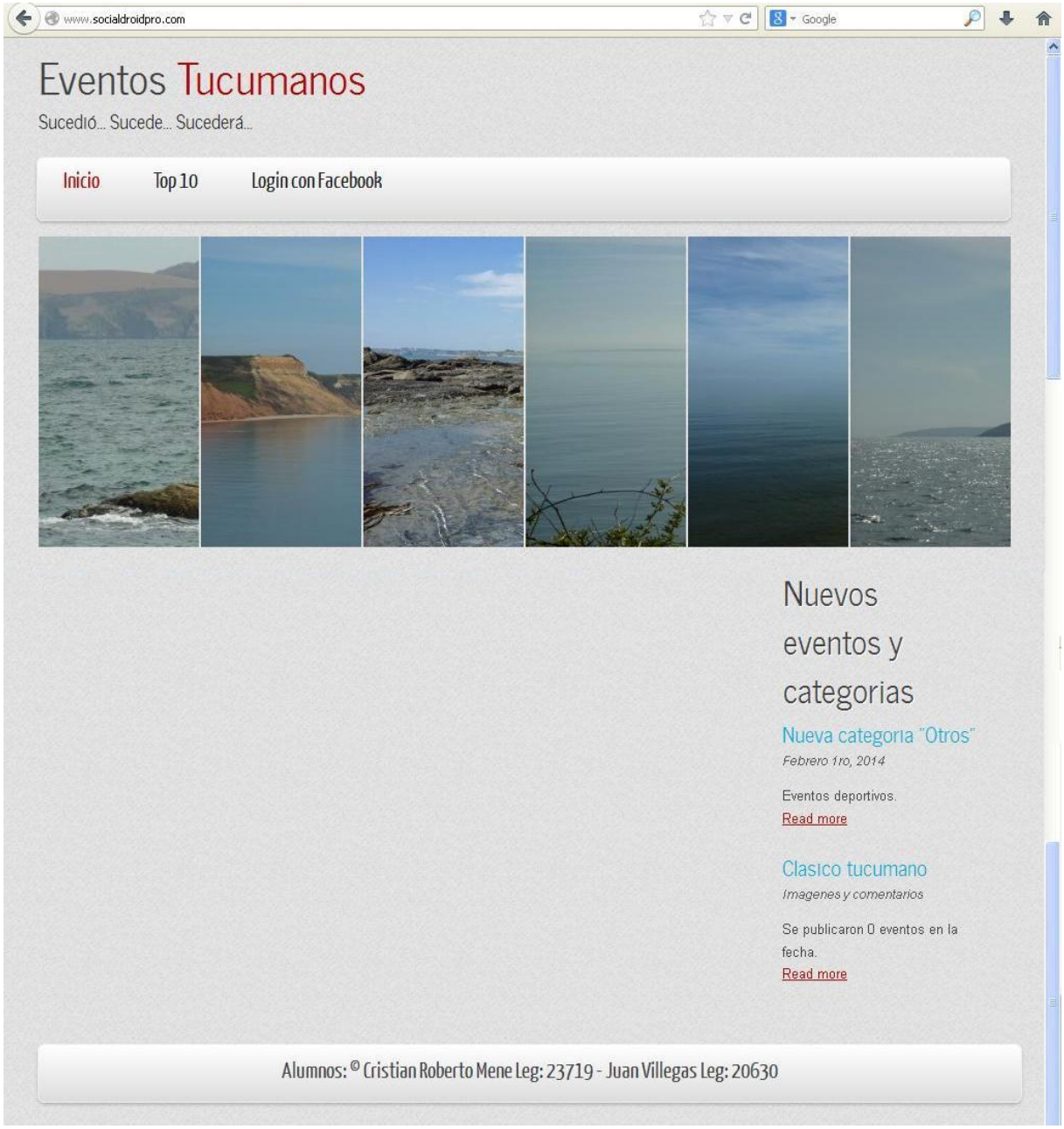
3.2 Diseño de la Aplicación:

3.2.1 Diseño de los componentes de la Aplicación:

Una aplicación Web con Base de Datos consta de formularios, consultas, reportes, menús y programas de aplicación. Utilizando Lenguaje PHP a través de Adobe Dreamweaver CS6 permite realizar el diseño de los mismos ya que posee la característica de ser un Lenguaje Visual.

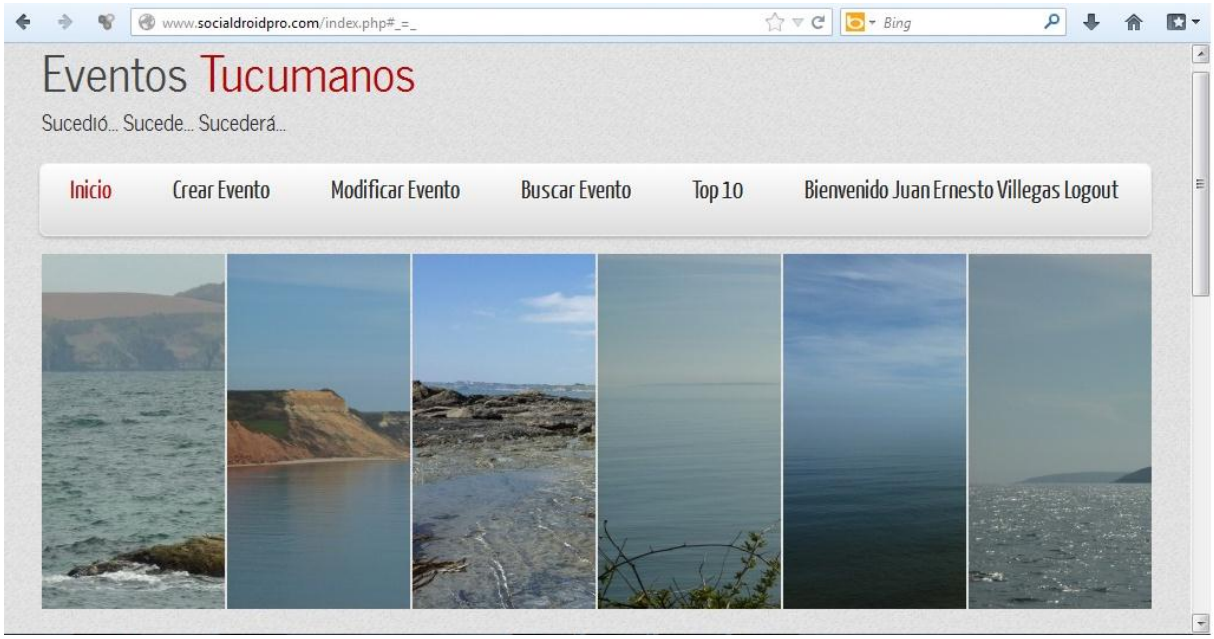
3.2.2 Diseño del Menú Principal de SocialDroid

Vista del Menú Principal de la Aplicación en modo de ejecución:



3.2.3 Módulos de Software necesarios:

Los módulos de software necesarios para esta aplicación son los que se muestran en el menú principal:



Formulario de Usuario Logueado

3.2.4 Diseño de las Formas de presentar los datos:

Para presentar los datos en pantalla se eligió los Formularios de entrada de datos. Para su diseño se utilizaron los siguientes componentes de Adobe Dreamweaver CS6:

- Formulario
- Cajas de texto
- Botones
- Grilla

3.2.5 Diseño de Consultas

Para su diseño se utilizaron los siguientes componentes de Adobe Dreamweaver CS6:

- Formulario
- Cajas de texto
- Grilla
- Botones

3.2.6 Diseño de Reportes

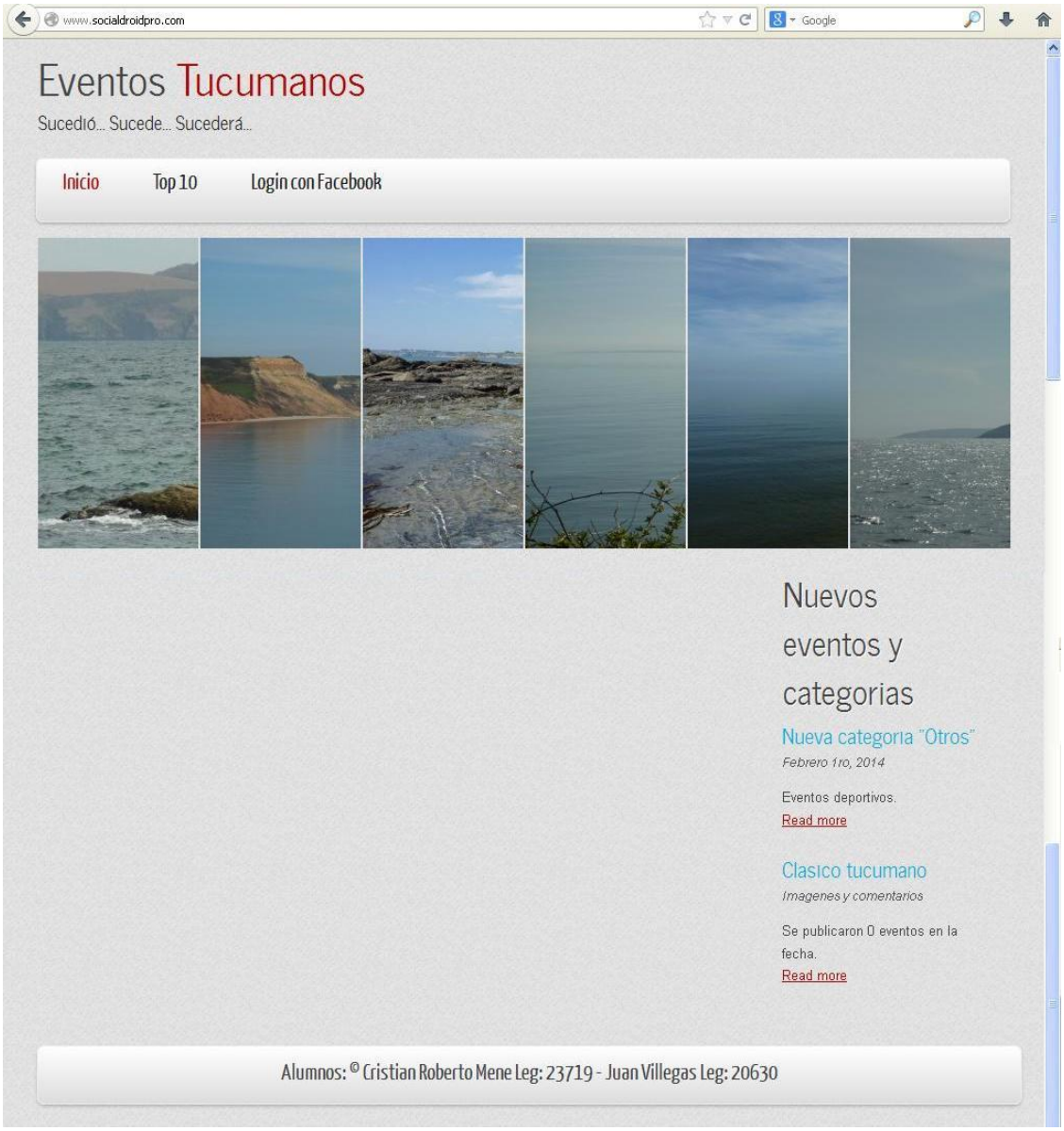
Se implementaron reportes web, utilizándose el Internet Explorer y HTML.

3.2.7 Diseño de Pantallas importantes

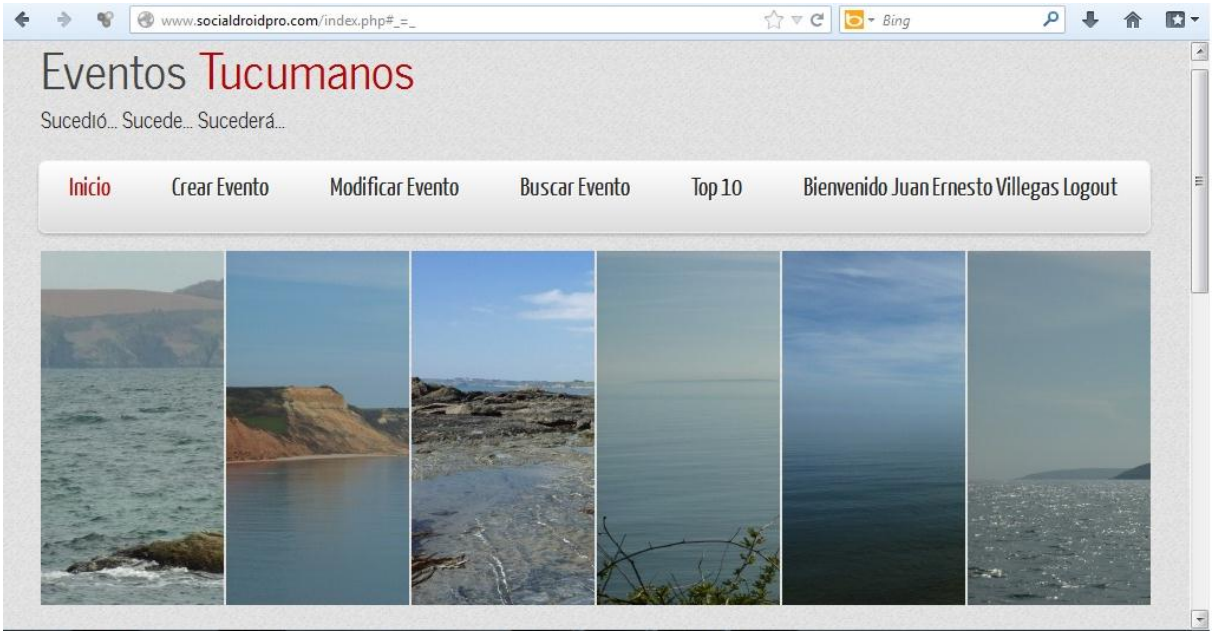
A continuación se muestra el diseño de pantallas más relevantes de la aplicación.

El código implementado en los distintos módulos se muestra en el punto de Programación del Sistema.

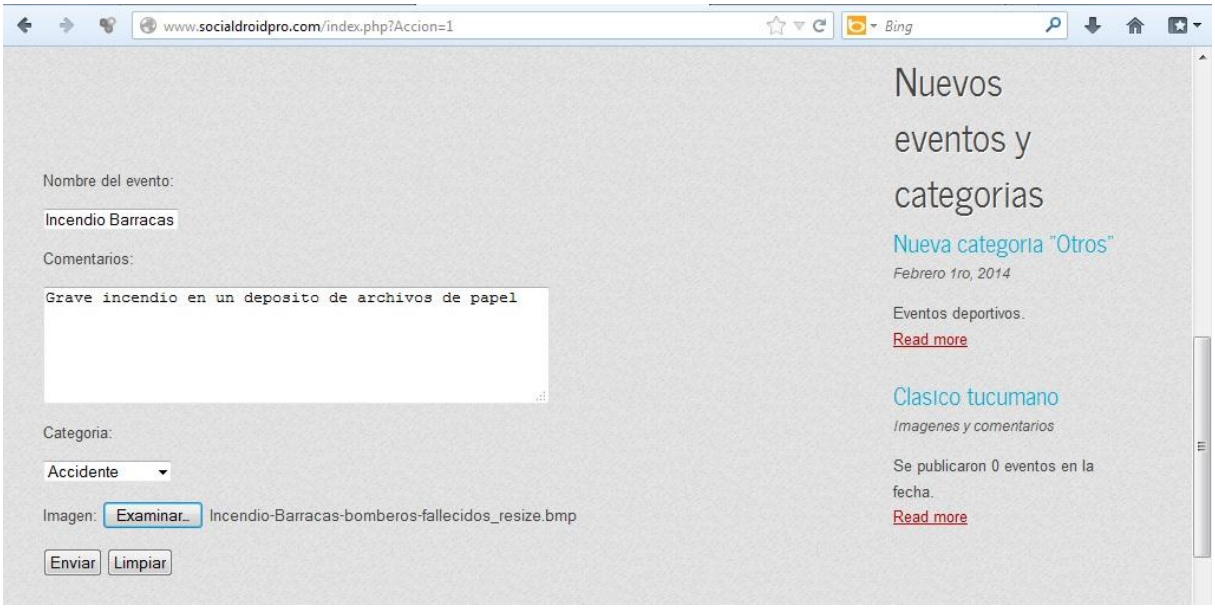
Pantallas:



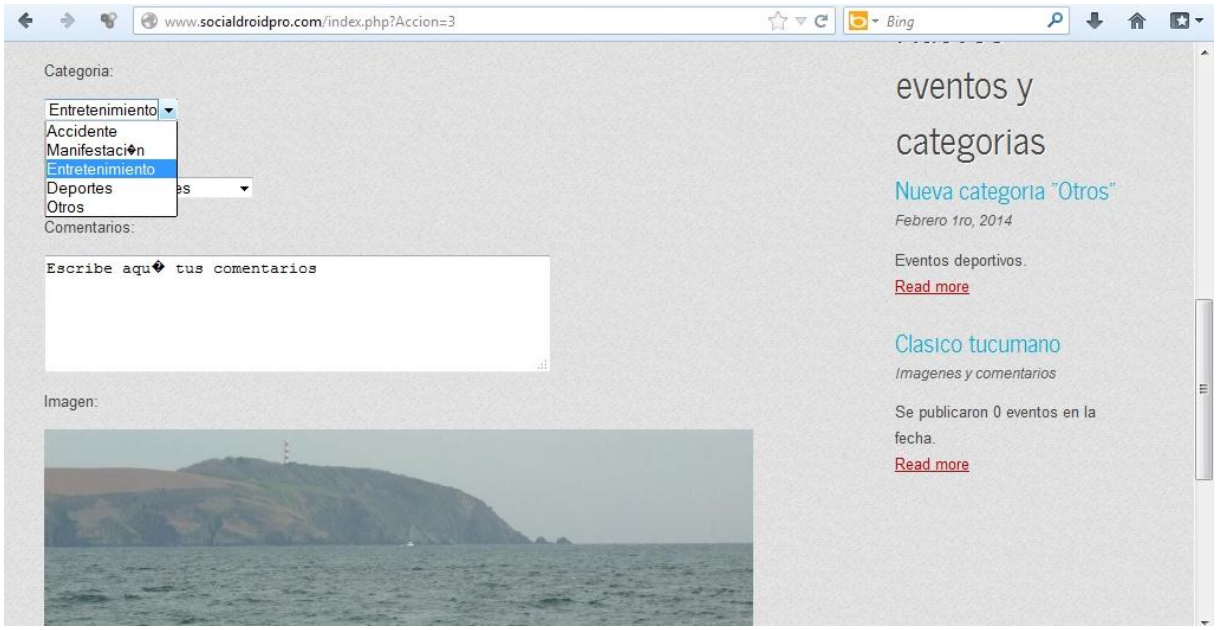
Pantalla principal



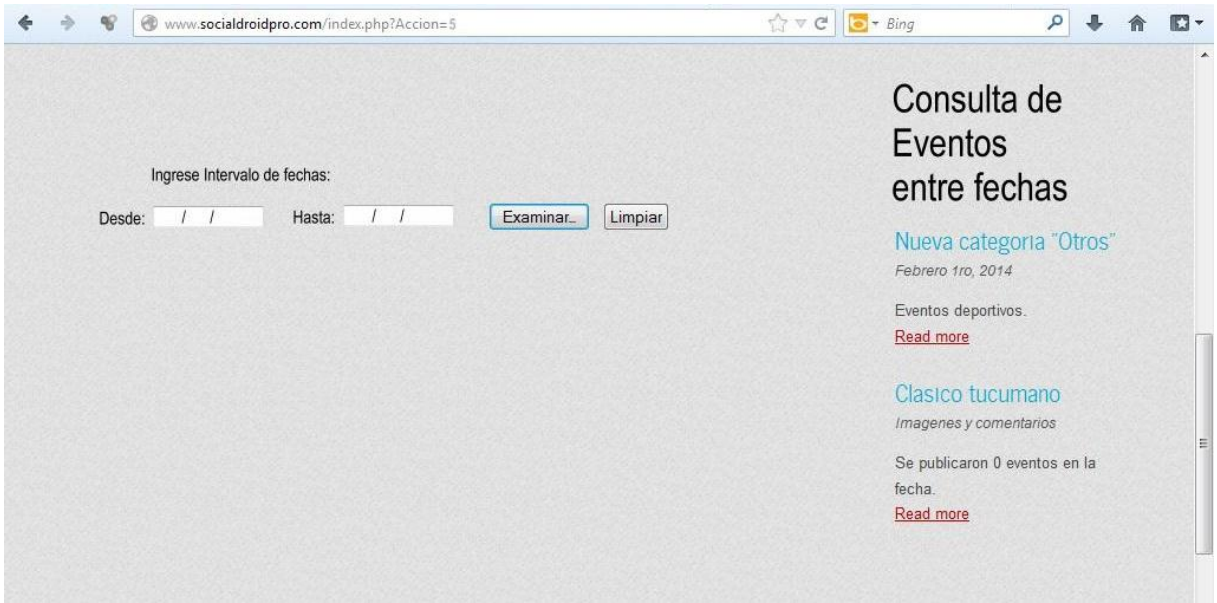
Formulario de Usuario Logueado



Formulario de Creación de eventos



Formulario de Búsqueda de eventos



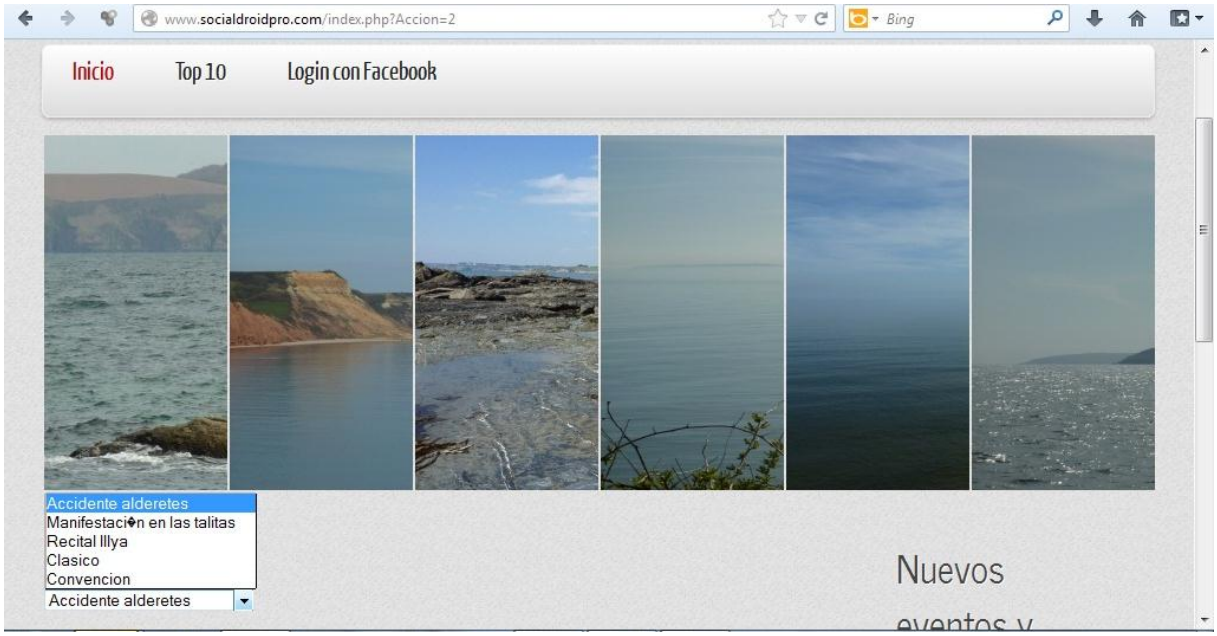
Formulario de Consulta de eventos entre fechas

RESULTADO DE LA CONSULTA

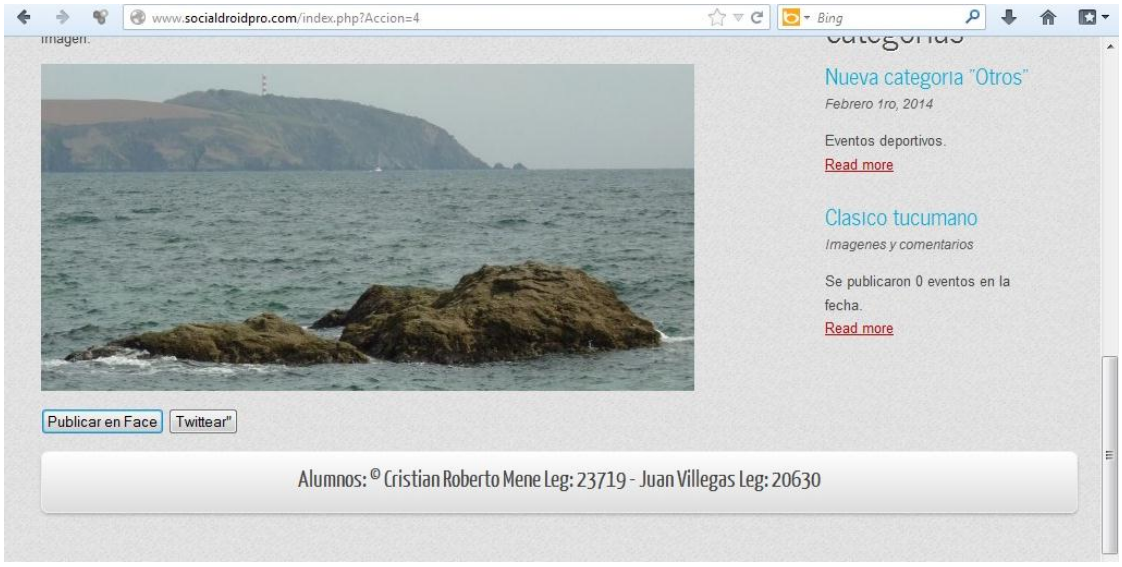
Listado de Eventos ocurridos entre:
el 04-02-2014 y el 06-02-2014

Fecha	Hora	Evento	Categoria	Dirección	Posición	NomUsuario
2014-02-04	16:30:00	Disturbios en Alderetes	Disturbios	Autopista Juan Domingo Perón	-26.824492, -65.1	Cristian Mene
2014-02-06	08:00:00	Incendio en Barracas	Incendio	Coronel Salvadores y Azara. Barracas	-34.644118, -58.3	Juan Villegas

Formulario de Reporte entre Rango de Fechas



Formulario de Modificación de eventos



Formulario que muestra cómo compartir eventos

4. Modelado a través de UML

Modelamos la situación problemática planteada mediante UML[6], Lenguaje Unificado de Construcción de Modelos, realizando todas las etapas con sus respectivas actividades para completar el primer ciclo de desarrollo.

4.1 Fase de Planeación y Elaboración

4.1.1 Definición de los Requerimientos

Los requerimientos son una descripción de las necesidades o deseos de un producto o negocio. La meta primaria de la fase de requerimientos es identificar y documentar lo que en realidad se necesita.

Se realizan los siguientes artefactos: Panorama General, Clientes, Metas, Funciones del Sistema y Atributos del Sistema.

4.1.1.1 Panorama General del Sistema

Este proyecto tiene por objeto crear un paquete de software para gestionar la integración de las redes sociales más populares del mundo aprovechando su utilización.

4.1.1.2 Clientes

Cualquier empresa que desee tener nuestro sistema y/ó entidades publicitarias que deseen incorporarse a nuestro sistema.

4.1.1.3 Metas

La meta incluye:

- ✓ Permitir la registración de accesos de usuarios de las redes sociales más populares en la herramienta.
- ✓ Permitir la publicación de diversos contenidos compartidos entre las redes sociales conectadas a la herramienta.
- ✓ Almacenar en una base de datos los datos proporcionados por los usuarios así como las estadísticas y resultados obtenidos luego del procesamiento de dichos datos.
- ✓ Emitir informes sobre el acceso a la herramienta para facilitar la venta del producto a los distintos posibles anunciantes de publicidad.
- ✓ Agilizar la acción de las entidades correspondientes ante la ocurrencia de un siniestro.

6 Bibliografía guía: UML y Patrones –Larman, Craig.

4.1.1.4 Funciones del sistema

Ref #	Función	Categoría
R1.1	Registra los datos de los usuarios.	evidente
R1.2	Registra los accesos de usuarios.	evidente
R1.3	Registra los eventos.	evidente
R1.4	Permite buscar un evento.	evidente
R1.5	Permite categorizar un evento.	evidente
R1.6	Permite asignar puntos a un evento.	evidente
R1.7	Permite compartir un evento.	evidente
R1.8	Permite comentar un evento.	evidente
R1.9	Permite cerrar la sesión de un usuario.	evidente
R1.10	Permite ubicar los usuarios en un ranking.	evidente
R1.11	Imprime el reporte de accesos de usuarios.	evidente
R1.12	El usuario deberá introducir una identificación y una contra-seña para poder utilizar el sistema.	evidente
R1.13	Ofrece un mecanismo de almacenamiento persistente.	oculta
R1.14	Permite conectar la aplicación desarrollada con la base de datos creada.	oculta
R1.15	Permite consultas a través de Internet.	evidente
R1.16	Permite conectar la aplicación desarrollada con la base de datos creada.	oculta
R1.17	Permite efectuar consultas de usuarios	evidente
R1.18	Maneja reportes web	evidente
R1.19	Permite Ver eventos en un mapa	evidente
R1.20	Ofrece ayuda a los usuarios	evidente

4.1.1.5 Atributos del sistema

Atributo	Detalles y restricciones de frontera
Tiempo de respuesta	(restricción de frontera) Los datos de un usuario deberán mostrarse en 3 segundos. (restricción de frontera) Los reportes deberán imprimirse en 10 segundos. (restricción de frontera) La ayuda deberá mostrarse en 2 segundos. (restricción de frontera) La página web deberá cargarse en 5 segundos.
Metáfora de interfaz	(detalle) Desarrollo del sistema en un lenguaje visual que posibilite la conexión con una base datos. (detalle) Permitir la utilización del mouse y el teclado.
Plataforma del sistema operativo	(detalle) Windows XP/Seven/8, Android
Facilidad de uso	(detalle) guiar al operador durante la utilización del sistema mediante la ayuda de la aplicación.

4.1.1.6 Casos de Uso

4.1.1.6.1 Actores identificados

- ✓ Usuario

4.1.1.6.2 Casos de uso identificados

Registrar usuario

Iniciar sesión

Cerrar sesión

Registrar evento

Buscar evento

Comentar evento

Compartir evento

Ubicar evento en mapa

Categorizar evento

Calificar evento

Ver eventos en mapa

4.1.1.6.3 Casos de uso en formato de alto nivel

Caso de Uso: Registrar usuario

Actores: Visitante.

Tipo: Secundario

Descripción: Un visitante ingresa al sitio web de SocialDroid y hace click en el botón registrarse, selecciona la red social con la cual quiere acceder (Facebook ó Twitter), luego ingresa su email y contraseña. Autoriza el uso de su cuenta en la aplicación. El sistema verifica con la red social si el visitante se encuentra registrado. El sistema captura los datos del nuevo usuario y los ingresa en la Base de Datos del sistema.

Caso de Uso: Iniciar sesión

Actores: Usuario.

Tipo: Secundario

Descripción: Un usuario ingresa su nombre de usuario y contraseña para tener acceso al sitio web de SocialDroid y hace clic en el botón iniciar sesión. El sistema captura los datos del usuario y los ingresa en la Base de Datos del sistema.

Caso de Uso: Cerrar sesión

Actores: Usuario.

Tipo: Secundario

Descripción: El usuario selecciona la opción cerrar sesión y sale de la aplicación SocialDroid. El sistema cierra la sesión del usuario.

Caso de Uso: Registrar evento

Actores: Usuario.

Tipo: Primario

Descripción: Un usuario logueado en SocialDroid, mediante su cuenta de Facebook (ó Twitter como se implementará a futuro), agrega un evento de forma proactiva mostrándose en el mapa para todos los usuarios.

Caso de Uso: Buscar evento

Actores: Usuario.

Tipo: Secundario

Descripción: Un usuario logueado en SocialDroid buscará un evento escribiendo el nombre del evento o la fecha de ocurrencia del mismo.

Caso de Uso: Comentar evento

Actores: Usuario.

Tipo: Secundario

Descripción: Un usuario logueado en SocialDroid podrá comentar un evento seleccionado.

Caso de Uso: Compartir evento

Actores: Usuario.

Tipo: Secundario

Descripción: Un usuario logueado en SocialDroid podrá compartir un evento seleccionado y este se divulgará a la red social con la que se encuentra logueado.

Caso de Uso: Ubicar evento en mapa

Actores: Usuario.

Tipo: Secundario

Descripción: Un usuario logueado en SocialDroid podrá visualizar en un mapa la ubicación de un evento.

Caso de Uso: Categorizar evento

Actores: Usuario.

Tipo: Secundario

Descripción: Un usuario logueado en SocialDroid podrá seleccionar la categoría a la cual pertenece un evento.

Caso de Uso: Calificar evento

Actores: Usuario.

Tipo: Secundario

Descripción: Un usuario logueado en SocialDroid podrá asignar puntos a un evento dado.

Caso de Uso: Ver eventos en mapa

Actores: Usuario.

Tipo: Primario

Descripción: Un usuario logueado en SocialDroid podrá observar en un mapa global todos los eventos ocurridos hasta el momento dentro del intervalo de fechas seleccionado. Pudiendo el usuario imprimir el reporte de eventos.

4.1.1.6.4 Casos de uso en formato expandido

Caso de Uso: Registrar evento

Actores: Usuario (Iniciador).

Propósito: Realizar la registración de un evento ocurrido recientemente.

Resumen: Un usuario logueado en SocialDroid, mediante su cuenta de Facebook (ó Twitter como se implementará a futuro), agrega un evento de forma proactiva mostrándose en el mapa para todos los usuarios.

Tipo: Primario y esencial.

Referencias Cruzadas: Funciones: R.1.3 y R.1.12

Curso normal de los eventos:

Acción del actor	Respuesta del sistema
1. Este caso de uso comienza cuando un Usuario logueado intenta registrar un evento en el sistema web SocialDroid.	2. Solicita al usuario que indique acerca del evento un comentario, una foto, la ubicación donde sucedió, fecha y hora.
3. El usuario ingresa un nombre de evento, un comentario, una foto, la ubicación donde sucedió, fecha y hora, y selecciona la opción publicar.	4. Publica los datos ingresados por el usuario.
5. El usuario puede visualizar el evento que publicó.	

Cursos alternos:

Línea 4: El sistema detecta contenido ofensivo o provocador. Indicar error.

Caso de Uso: Ver evento en mapa

Actores: Usuario.

Propósito: Observar lugares donde ocurrieron eventos en el transcurso del día.

Resumen: Un usuario logueado en SocialDroid podrá observar en un mapa global todos los eventos ocurridos hasta el momento dentro del intervalo de fechas seleccionado para los cuales se indicó la ubicación del evento de los eventos, observando fecha y hora de ocurrencia de los mismos.

Tipo: Primario y esencial.

Referencias Cruzadas: Funciones: R.1.19

Curso normal de los eventos:

Acción del actor	Respuesta del sistema
1. Este caso de uso comienza cuando un usuario ingresa un intervalo de fechas y selecciona la opción visualizar mapa de eventos.	2. Muestra en pantalla un mapa global con los eventos ocurridos en el intervalo de fechas seleccionado.

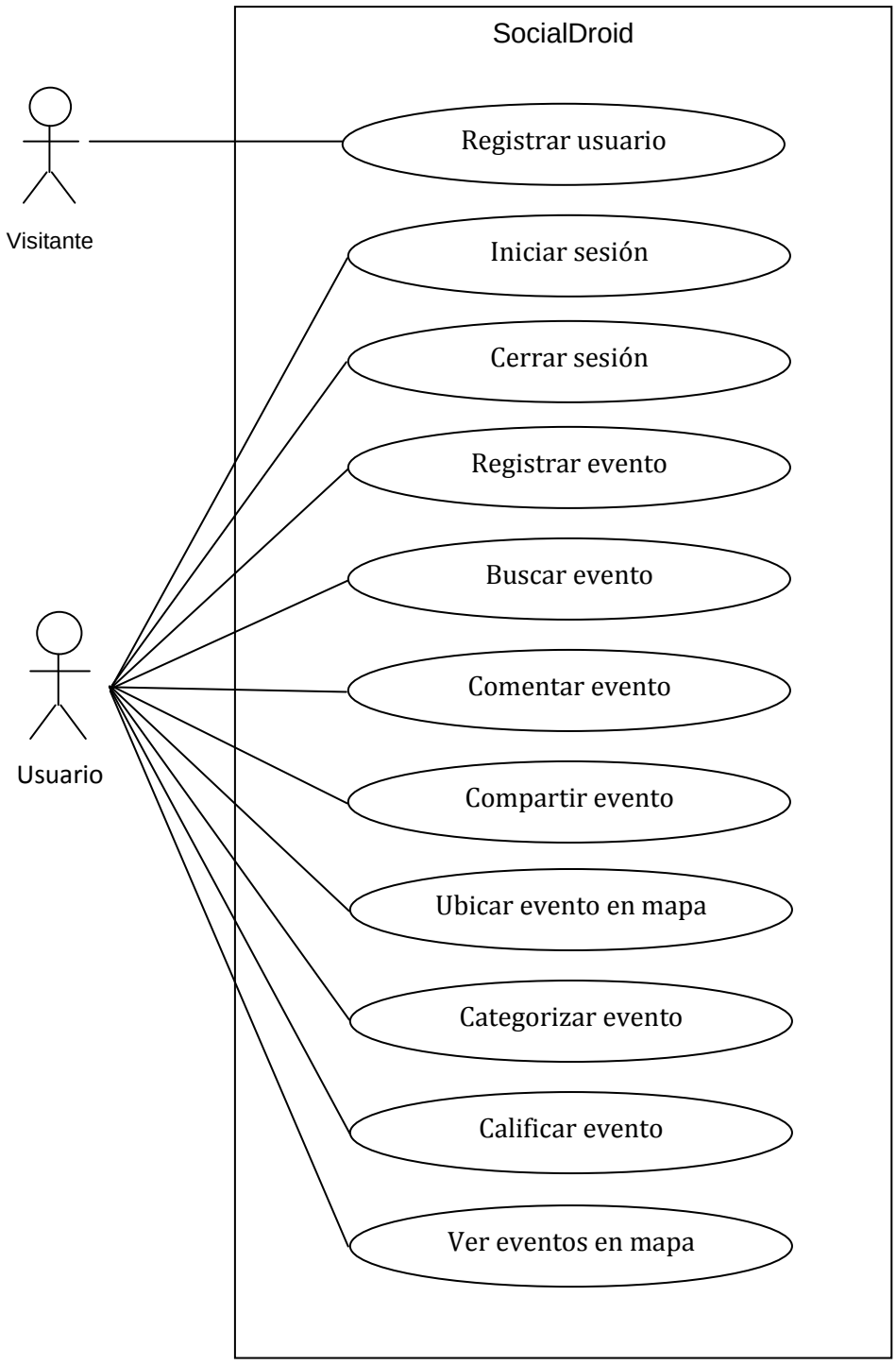
3. El usuario podrá visualizar en el mapa el lugar, fecha y hora en el que ocurrió un evento al posicionarse sobre el mismo. Al pulsar el botón imprimir le indica a la Terminal que imprima el reporte con los eventos ocurridos.	4. Genera e imprime el reporte.
5. El usuario tiene impreso el reporte de los eventos ocurridos.	

Cursos alternos:

Línea 2: No existen eventos registrados en el día de la fecha. Indicar error.

Línea 4: La impresora no está lista o no hay papel. Indicar error.

4.1.1.6.5 Diagrama de casos de uso

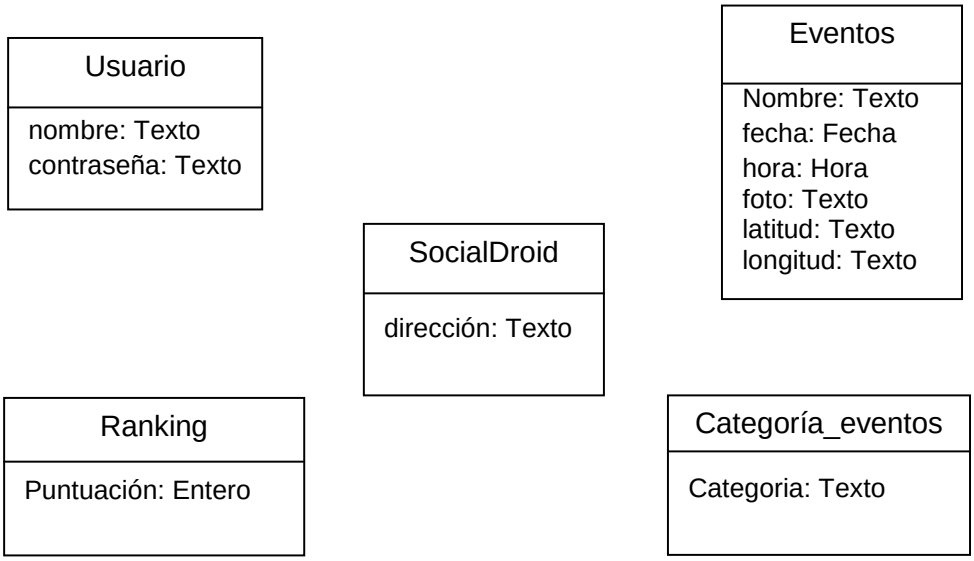


4.2 Fase de Construcción: Inicio del primer Ciclo de Desarrollo

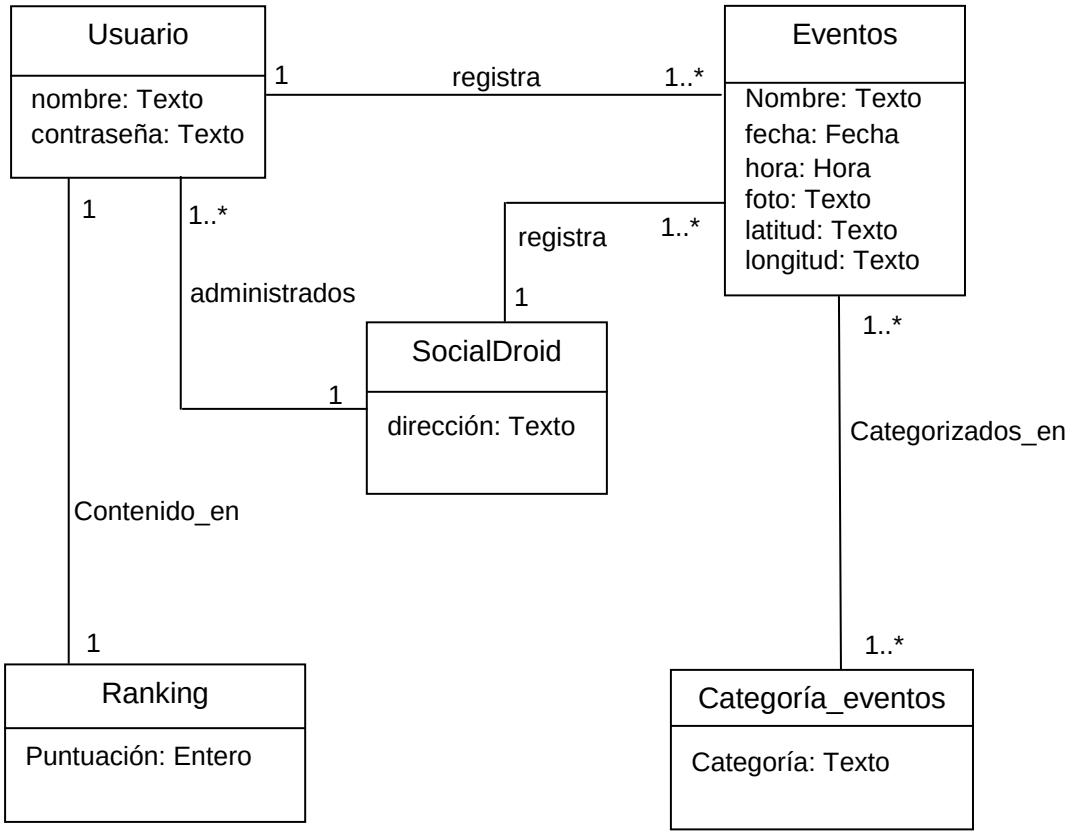
4.2.1 Fase de Análisis

4.2.1.1 Modelo conceptual

4.2.1.1.1 Conceptos y Atributos identificados



4.2.1.1.2 Modelo conceptual

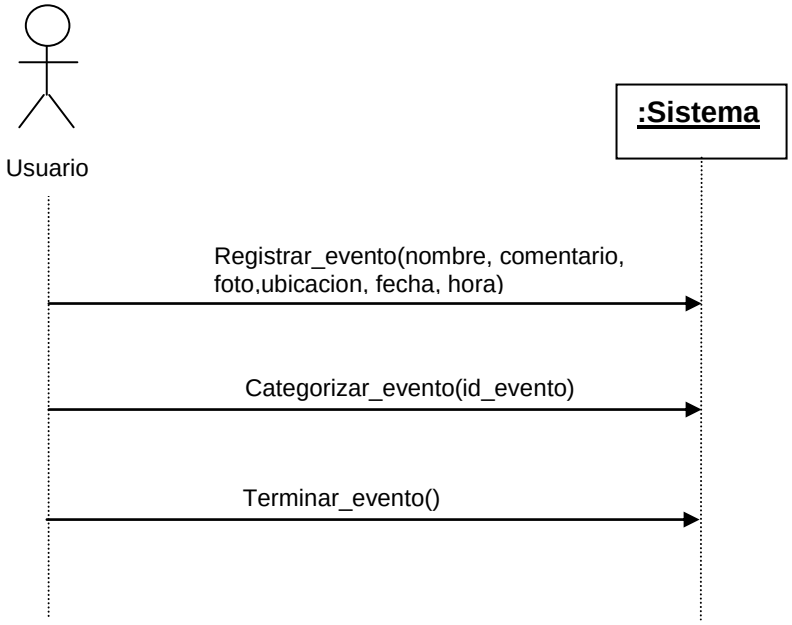


4.2.1.1.3 Elaboración del Glosario

Término	Categoría	Comentarios
Categorizar evento	Caso de Uso	Descripción del proceso que efectúa un usuario para discriminar un evento de otro.
Eventos.fecha: fecha	Atributo	Fecha en que se efectuó el evento.
Eventos.hora: hora	Atributo	Hora en que se efectuó el evento.
Eventos.nombre: texto	Atributo	Nombre asignado al evento.
Eventos.foto: texto	Atributo	Nombre de la imagen capturada en el lugar del evento.
Eventos.latitud: texto	Atributo	Latitud del lugar del evento.
Eventos.longitud: texto	Atributo	Longitud del lugar del evento.
Categoría evento	Concepto	Distintas secciones a las que puede pertenecer un evento.
Usuario	Concepto	Persona que requiere las prestaciones de SocialDroid.
Registrar evento	Caso de Uso	Descripción del proceso mediante el cual un usuario agrega un evento al sistema.
Ranking	Concepto	Sección donde se asignan puntos al usuario por los eventos registrados.
Eventos	Concepto	Lugar que contiene toda la información de un evento registrado.
SocialDroid	Concepto	Un lugar donde se realizan los registros de eventos y se administran.
Usuario.nombre: texto	Atributo	Nombre del usuario que realiza la administración de eventos.
Usuario.contraseña: texto	Atributo	Clave del usuario que realiza la administración de eventos
Ranking.puntuacion: Entero	Atributo	Valor asignado al usuario por los eventos registrados.

4.2.1.2 Diagramas de la secuencia del sistema

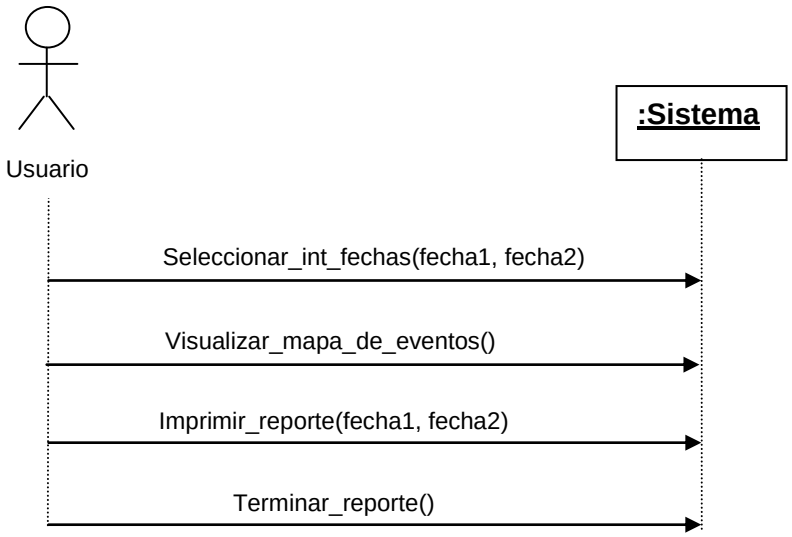
4.2.1.2.1 Diagrama de la secuencia para el caso de uso: Registrar evento.



Operaciones del sistema:

Sistema
Registrar_evento(nombre, comentario, foto, ubicacion, fecha, hora) Categorizar_evento(id_evento) Terminar_evento()

4.2.1.2.2 **Diagrama de la secuencia para el caso de uso:** Ver evento en mapa



Operaciones del sistema:

Sistema
Seleccionar_int_fechas(fecha1, fecha2) Visualizar_mapa_de_eventos() Imprimir_reporte(fecha1, fecha2) Terminar_reporte()

4.2.1.3 Contratos de operaciones

4.2.1.3.1 **Contratos para el caso de uso:** Registrar evento

4.2.1.3.1.1 **Contrato para la operación**: Registrar evento

Nombre:	Registrar_evento(nombre, comentario, foto, ubicacion, fecha, hora)
Responsabilidades:	Capturar el comentario, foto, ubicacion, fecha y hora del evento a publicar de un usuario. Desplegar sus datos en pantalla. Realizar el registro del evento.
Tipo:	Sistema

Referencias Cruzadas:	Funciones del sistema: R.1.3, R.1.5. Caso de uso: Registrar evento.
Notas:	Utilizar el acceso a la base de datos.
Excepciones:	Si el comentario posee contenido ofensivo o provocador. Indicar que hay error.
Salida:	
Precondiciones	El sistema conoce el email y la contraseña del usuario.
Poscondiciones:	Se creó una instancia Registrar_evento (creación de una instancia). Se asoció la instancia Registrar_evento con Categorizar_evento (asociación formada).

4.2.1.3.1.2 **Contrato para la operación:** Categorizar_evento

Nombre:	Categorizar_evento(id_evento)
Responsabilidades:	Capturar el código de un evento y agregarlo a la actual transacción, desplegando una lista de las categorías existentes, permitiendo al usuario asignar al evento una categoría.
Tipo:	Sistema
Referencias Cruzadas:	Caso de uso: Registrar evento.
Notas:	Utilizar el acceso a la base de datos.
Excepciones:	Si el código del evento no existe, indicar error.
Salida:	
Precondiciones	El sistema conoce el código del evento. El sistema conoce la lista de categorías de eventos.
Poscondiciones:	Se asoció una instancia Eventos a una Categoría_eventos (asociación formada). Se asoció una instancia Eventos a una SocialDroid (asociación formada).

4.2.1.3.1.3 **Contrato para la operación:** Terminar_evento

Nombre:	Terminar_evento()
Responsabilidades:	Registrar el evento en un registro histórico de eventos.
Tipo:	Sistema
Referencias Cruzadas:	Caso de uso: Registrar evento.
Notas:	
Excepciones:	
Salida:	
Precondiciones	
Poscondiciones:	Se asignó a Evento.fecha la fecha actual (modificación de atributo). Se asignó a Evento.hora la hora actual (modificación de atributo). Se asoció una instancia Evento a SocialDroid (asociación formada).

4.2.1.3.1.4 **Contratos para el caso de uso**: Ver evento en mapa

4.2.1.3.1.5 **Contrato para la operación: Seleccionar_int_fechas**

Nombre:	Seleccionar_int_fechas(fecha1, fecha2)
Responsabilidades:	Seleccionar los eventos publicados según el período ingresado. Desplegar en pantalla nombre, comentario, foto, fecha, hora, latitud y longitud de ocurrencia del mismo. Mostrar la fecha de realización de la prestación.
Tipo:	Sistema.
Referencias Cruzadas:	Caso de uso: Ver evento en mapa.
Notas:	Utilizar el acceso a la base de datos.
Excepciones:	Si las fechas no son válidas, indicar error.
Precondiciones	El sistema conoce los eventos realizados por fecha.

Poscondiciones:	Se creó una instancia Visualizar_mapa_de_eventos (creación de una instancia).

4.2.1.3.1.6 **Contrato para la operación:** Visualizar_mapa_de_eventos

Nombre:	Visualizar_mapa_de_eventos ()
Responsabilidades:	Imprimir en pantalla el reporte de eventos ocurridos según el período especificado.
Tipo:	Sistema
Referencias Cruzadas:	Caso de uso: Ver evento en mapa.
Salida:	
Precondiciones	El sistema conoce los eventos realizados por fecha.
Poscondiciones:	Se creó la instancia Imprimir_reporte (fecha1, fecha2)

4.2.1.3.1.7 **Contrato para la operación:** Imprimir_reporte

Nombre:	Imprimir_reporte (fecha1, fecha2)
Responsabilidades:	Imprimir el reporte de eventos ocurridos según el período especificado.
Tipo:	Sistema
Referencias Cruzadas:	Caso de uso: Ver evento en mapa.
Salida:	Reporte de eventos ocurridos según el período especificado
Poscondiciones:	

4.2.1.3.1.8 **Contrato para la operación:** Terminar_reporte

Nombre:	Terminar_reporte()
Responsabilidades:	Registrar el reporte impreso en un registro histórico de reportes.

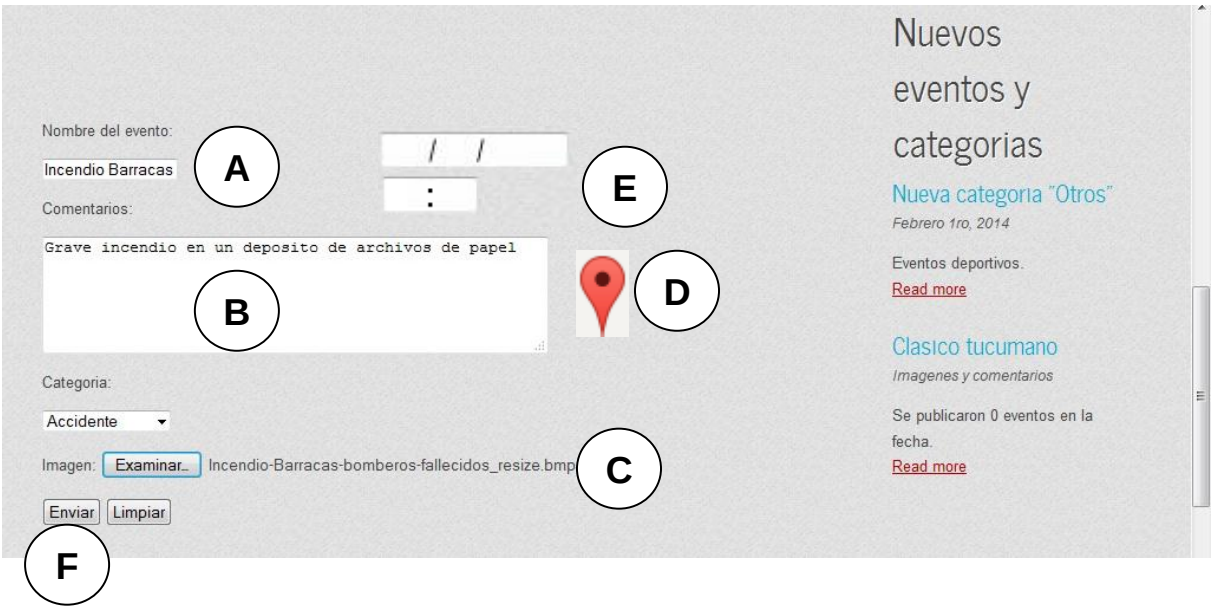
Tipo:	Sistema
Referencias Cruzadas:	Caso de uso: Ver evento en mapa.
Notas:	
Excepciones:	
Salida:	
Precondiciones	
Poscondiciones:	Se asignó a Reporte.fecha la fecha actual (modificación de atributo). Se asignó a Reporte.hora la hora actual (modificación de atributo). Se asoció una instancia Reporte a SocialDroid (asociación formada). Se asignó a Reporte.impreso el valor verdadero (modificación de atributo).

4.2.2 Fase de Diseño

4.2.2.1 Casos Reales de uso

4.2.2.1.1 **Caso de uso:** Registrar evento**.**

Interfaz Gráfica asociada



Caso de Uso: Registrar evento

Actores: Usuario (Iniciador).

Propósito: Realizar la registración de un evento ocurrido recientemente.

Resumen: Un usuario logueado en SocialDroid, mediante su cuenta de Facebook (ó Twitter como se implementará a futuro), agrega un evento de forma proactiva mostrándose en el mapa para todos los usuarios.

Tipo: Primario y real.

Referencias Cruzadas: Funciones: R.1.3 y R.1.12

Curso normal de los eventos:

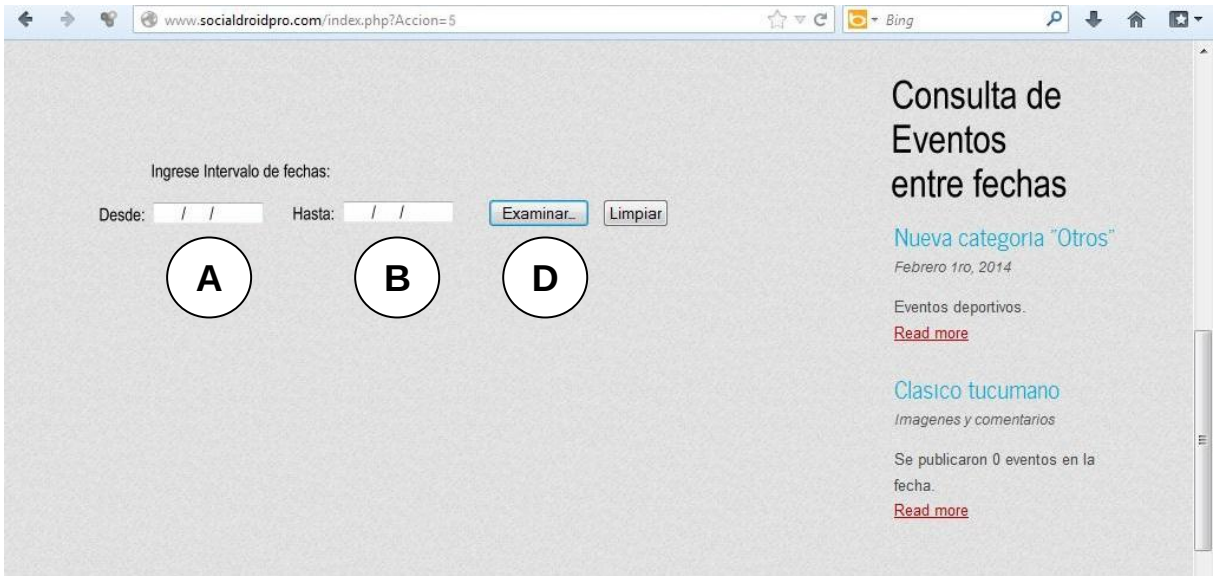
Acción del actor	**Respuesta del sistema**
1. Este caso de uso comienza cuando un Usuario logueado intenta registrar un evento en el sistema web SocialDroid.	
2. El usuario ingresa en la interfaz gráfica un nombre en **A**, un comentario en **B**, una foto en **C**, la ubicación donde sucedió en **D**, fecha y hora en **E**.	3. Completa los datos en la interfaz gráfica.
4. El usuario le indica a la Terminal que publique el evento creado accionando el botón **F**.	5. Registra el evento creado.
6. El usuario ha finalizado la publicación del evento.	

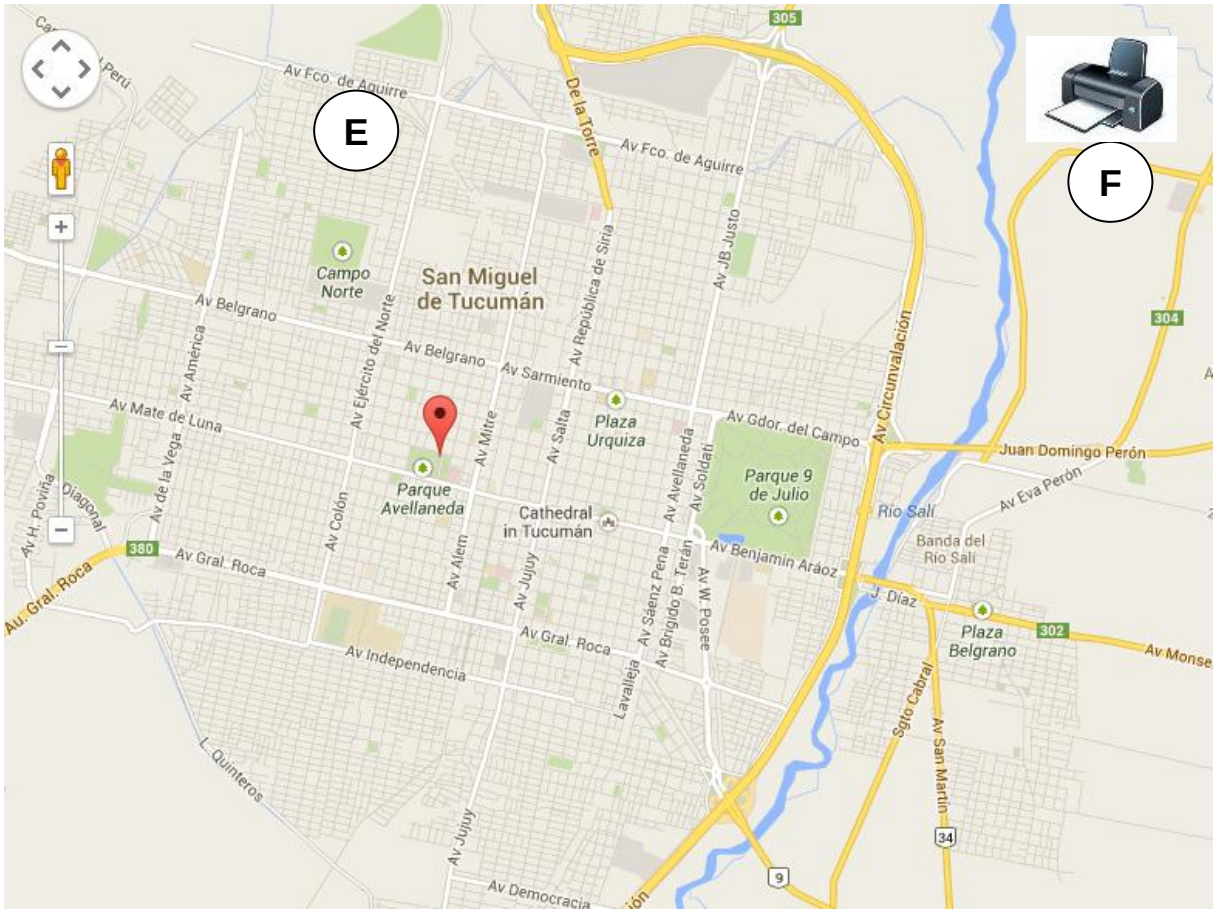
Cursos alternos:

Línea 2: El sistema detecta contenido ofensivo o provocador, ó identificador inválido. Indicar error.

4.2.2.1.2 **Caso de uso:** Ver evento en mapa**.**

Interfaz Gráfica asociada





RESULTADO DE LA CONSULTA

Listado de Eventos ocurridos entre:
el 04-02-2014 y el 06-02-2014

Fecha	Hora	Evento	Categoría	Dirección	Posición	NomUsuario
2014-02-04	16:30:00	Disturbios en Alderetes	Disturbios	Autopista Juan Domingo Perón	-26.824492, -65.1	Cristian Mene
2014-02-06	08:00:00	Incendio en Barracas	Incendio	Coronel Salvadores y Azara. Barracas	-34.644118, -58.3	Juan Villegas

Caso de Uso: Ver evento en mapa

Actores: Usuario.

Propósito: Observar lugares donde ocurrieron eventos en el transcurso del día.

Resumen: Un usuario logueado en SocialDroid podrá observar en un mapa global todos los eventos ocurridos hasta el momento dentro del intervalo de fechas seleccionado para los cuales se indicó la ubicación del evento de los eventos, observando fecha y hora de ocurrencia de los mismos.

Tipo: Primario y real.

Referencias Cruzadas: Funciones: R.1.19

Curso normal de los eventos:

Acción del actor	Respuesta del sistema
1. Este caso de uso comienza cuando un usuario ingresa un intervalo de fechas en **A** y **B**, y luego selecciona la opción visualizar mapa de eventos oprimiendo el botón **C**.	2. Selecciona los eventos ocurridos en el intervalo de fechas seleccionado. Completa la grilla en **D** con la información asociada. Muestra en pantalla un mapa global en **E** con los eventos.
3. El usuario al pulsar el botón imprimir en **F** le indica a la Terminal que imprima el reporte con los eventos ocurridos.	4. Genera e imprime el reporte.

5. El usuario tiene impreso el reporte de los eventos ocurridos.	

Cursos alternos:

Línea 1: El usuario ingresa un período inválido. Indicar error.

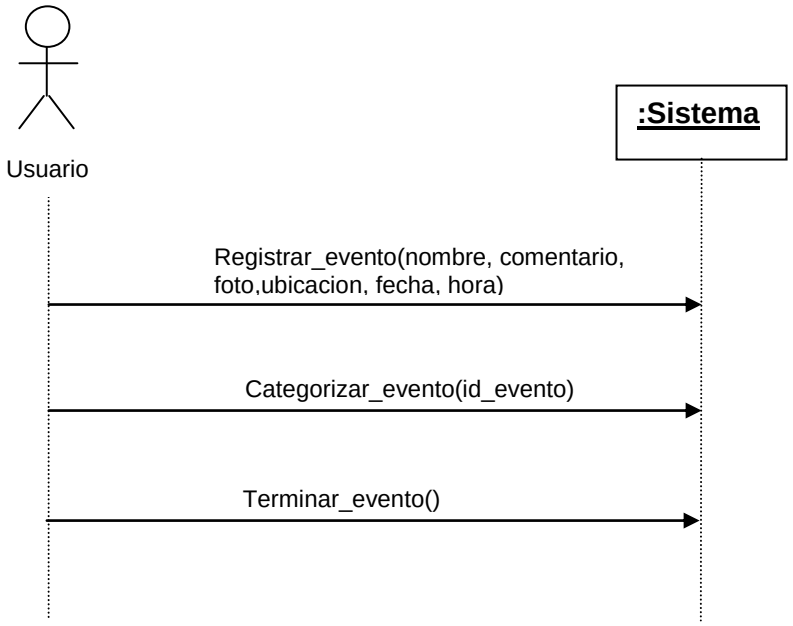
Línea 2: No existen eventos registrados en el día de la fecha. Indicar error.

Línea 4: La impresora no está lista o no hay papel. Indicar error.

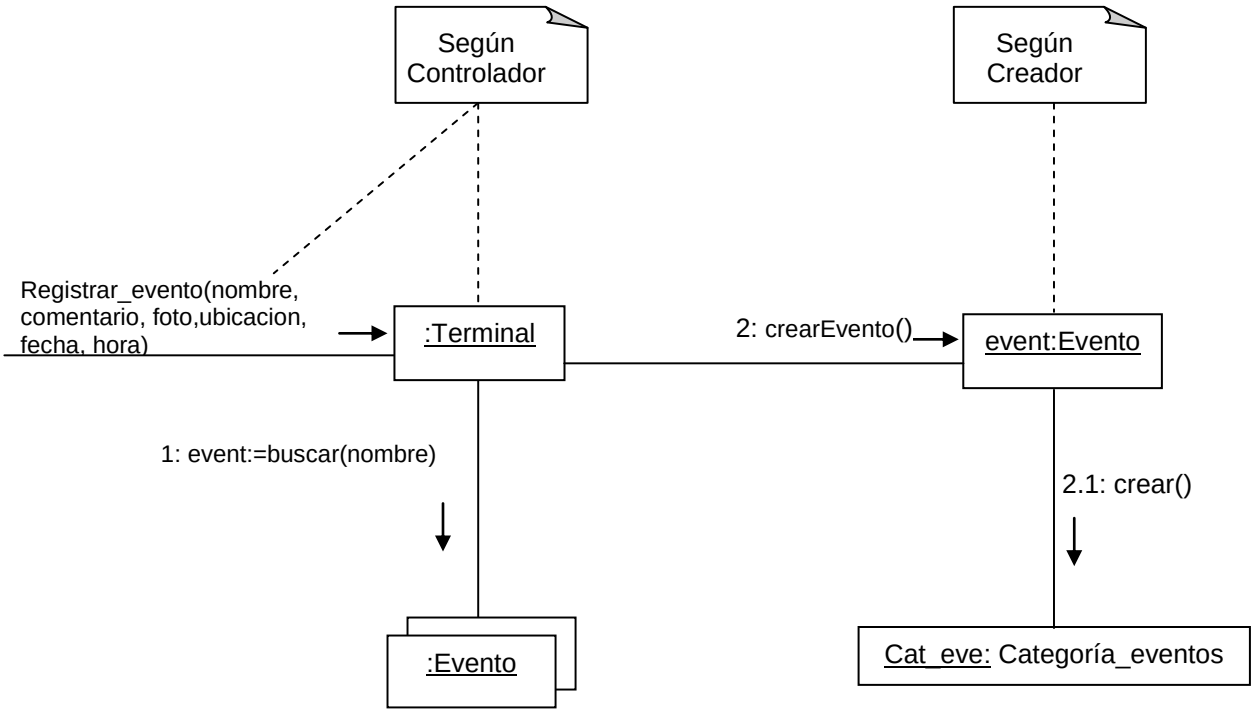
4.2.2.2 Diagramas de colaboración

4.2.2.2.1 **Caso de uso:** Registrar evento

4.2.2.2.1.1 **Diagrama de la secuencia para el caso de uso:** Registrar evento.



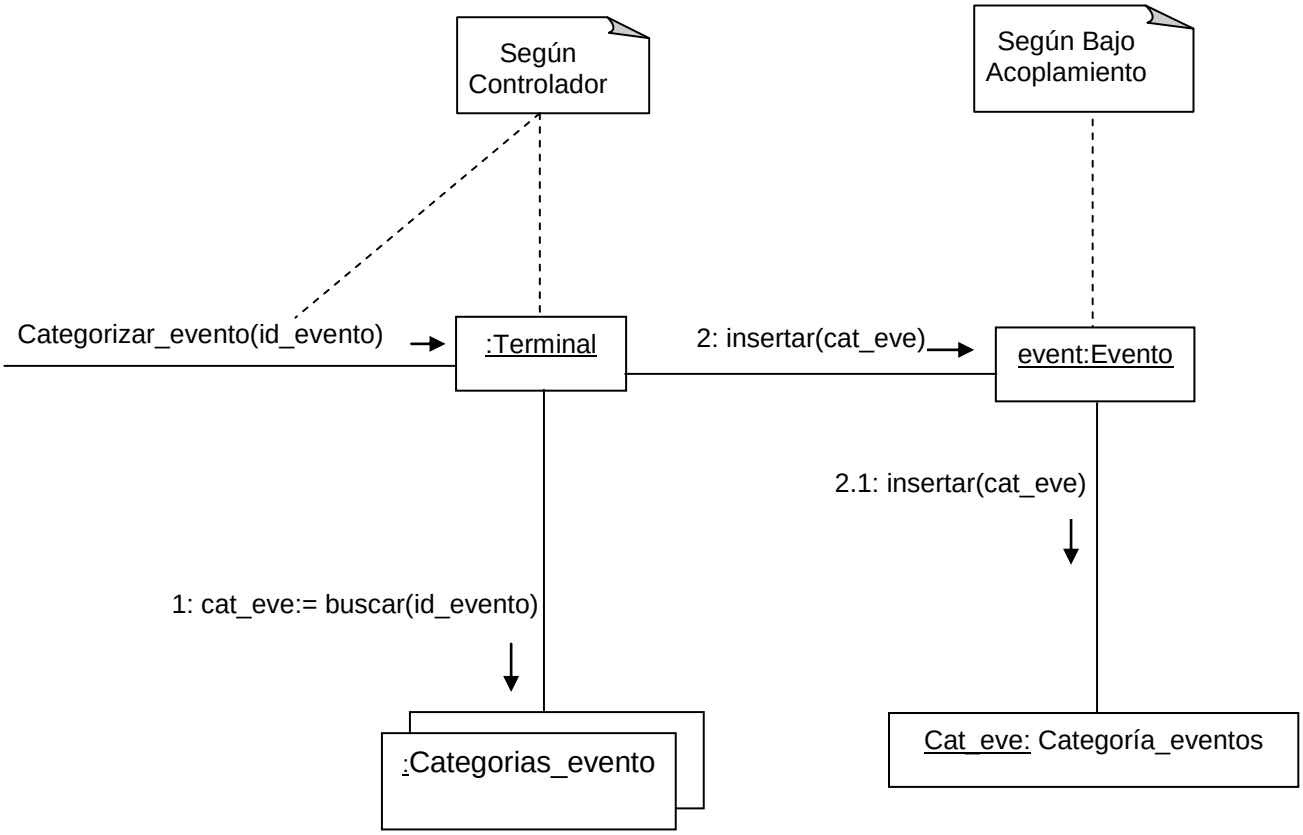
Evento: Registrar_evento



Responsabilidades:

Clase	Responsabilidad
Terminal	Encargada de manejar el evento de entrada Registrar_evento (Controlador de Fachada)
Evento	Encargada de crear las instancias de la clase Categoría_eventos (Creador)

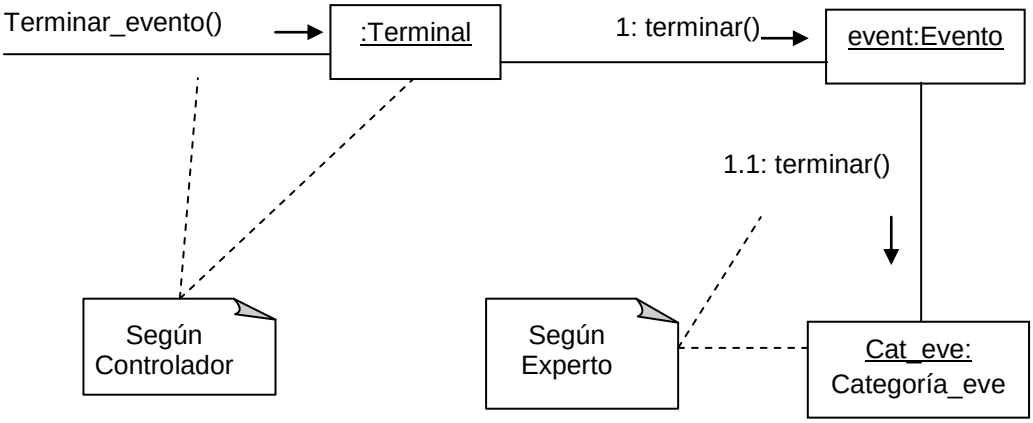
Evento: Categorizar_evento



Responsabilidades

Clase	Responsabilidad
Terminal	Encargada de manejar el evento de entrada Categorizar_evento (Controlador de Fachada)
Evento	Encargada de insertar los datos de la Categoría de eventos en el evento actual para que Terminal tenga un Bajo Acoplamiento.

Evento: Terminar_Evento



Responsabilidades

Clase	Responsabilidad
Terminal	Encargada de manejar el evento de entrada terminar (Controlador de Fachada)
Evento	Clase experto en conocer sus atributos fecha y hora

4.2.2.2.1.2 Diagrama de la secuencia para el caso de uso: Ver evento en mapa

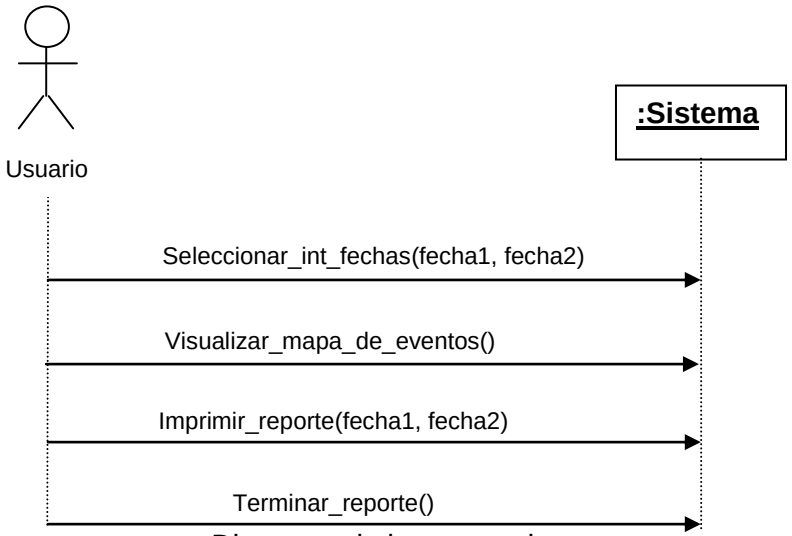
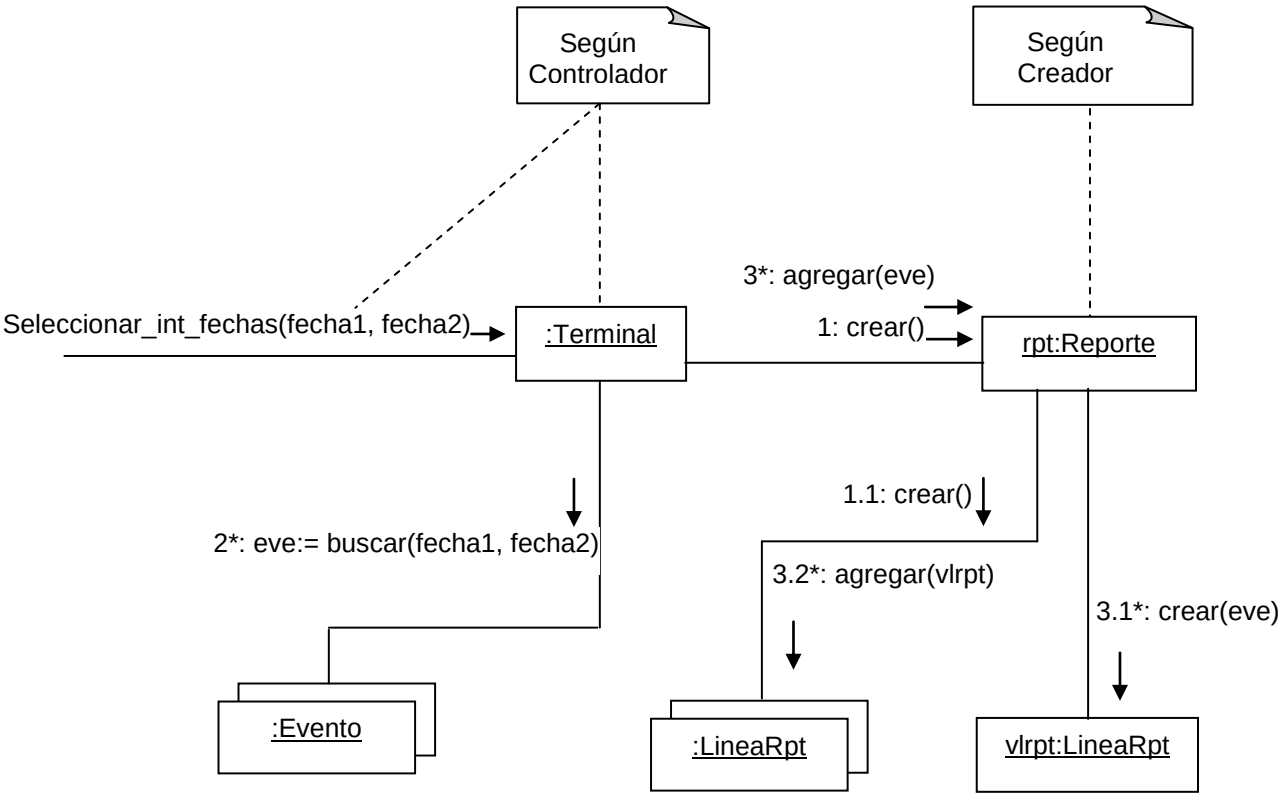


Diagrama de la secuencia

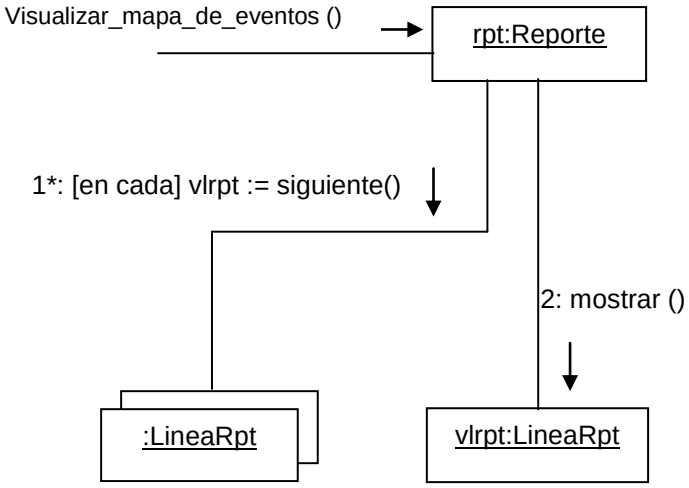
Evento: Seleccionar_int_fechas



Responsabilidades

Clase	Responsabilidad
Terminal	Encargada de manejar el evento de entrada Seleccionar_int_fechas (Controlador de Fachada)
Reporte	Encargada de crear las instancias de la clase LineaRpt (Creador)

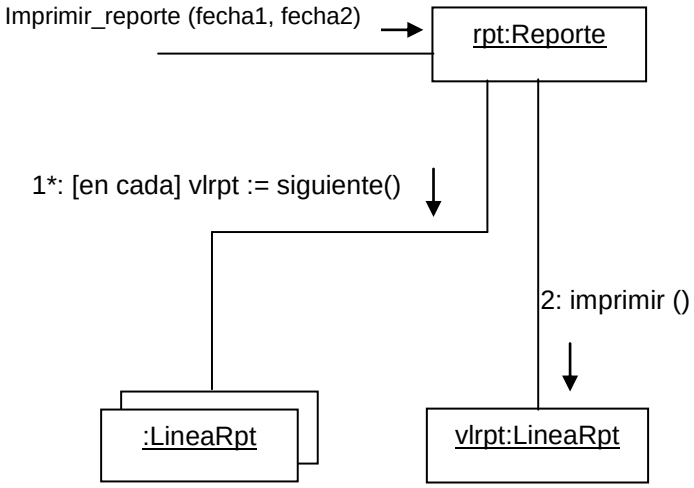
Evento: Visualizar_mapa_de_eventos



Responsabilidades

Clase	Responsabilidad
Reporte	Es el experto de información (Experto)
LineaRpt	Es el experto de información (Experto)

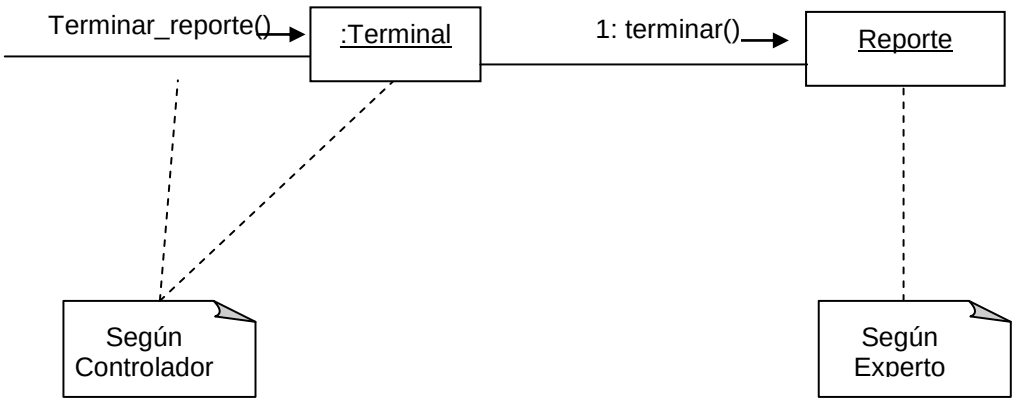
Evento: Imprimir_reporte



Responsabilidades

Clase	Responsabilidad
Reporte	Es el experto de información (Experto)
LineaRpt	Es el experto de información (Experto)

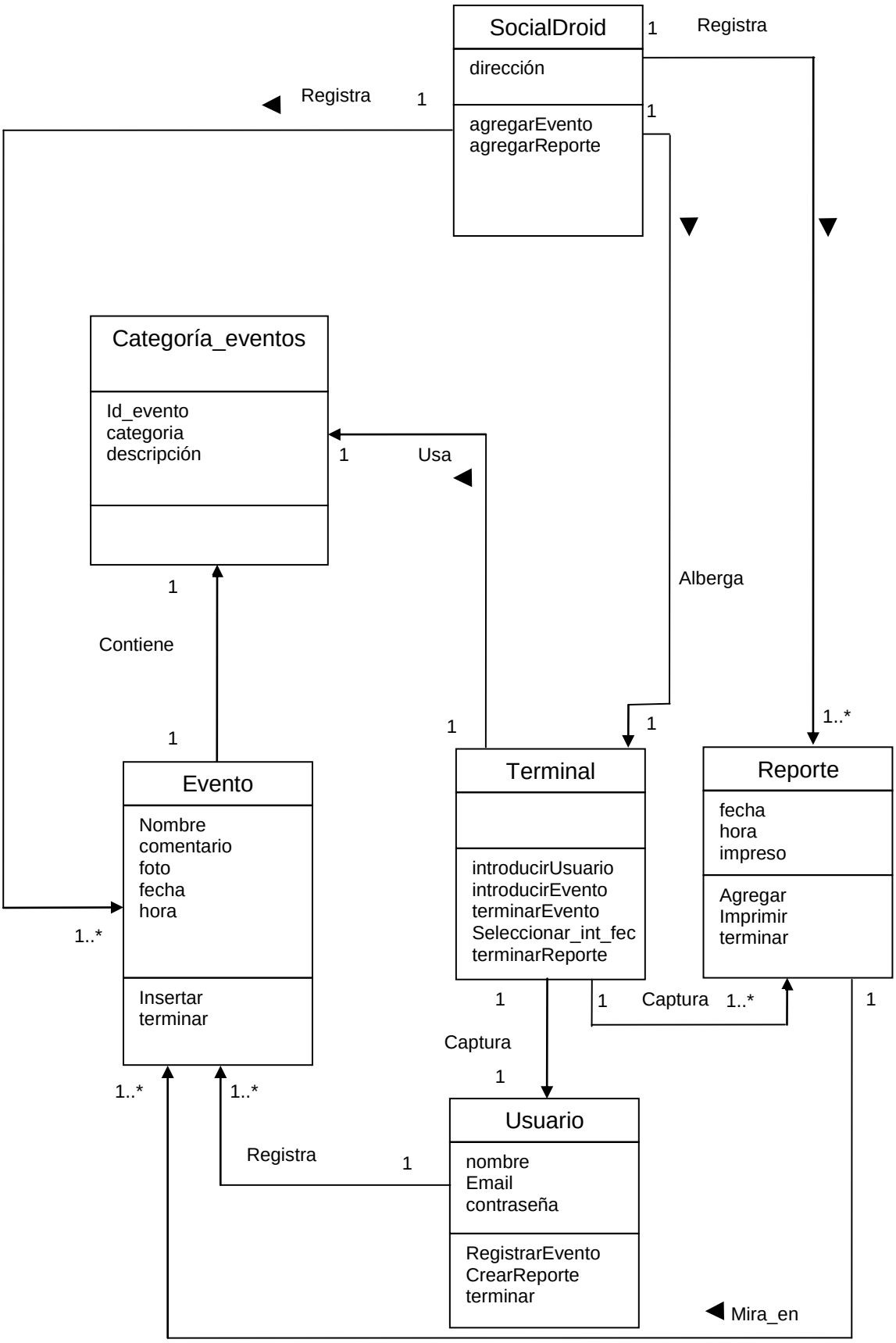
Evento: terminar_reporte



Responsabilidades

Clase	Responsabilidad
Terminal	Encargada de manejar el evento de entrada terminar_reporte (Controlador de Fachada)
Reporte	Clase **experto** en conocer sus atributos fecha, hora e impreso

4.2.2.3 Diagramas de Clases del diseño



5. Determinación de la metodología más conveniente

5.1 Introducción

La Metodología de Análisis y Diseño de Sistemas más conveniente que se debe aplicar en la situación problemática planteada es la del Diseño de Bases de Datos mediante el Modelo Entidad Relación y el Modelo Relacional con sus requerimientos, justificando la elección en los siguientes puntos:

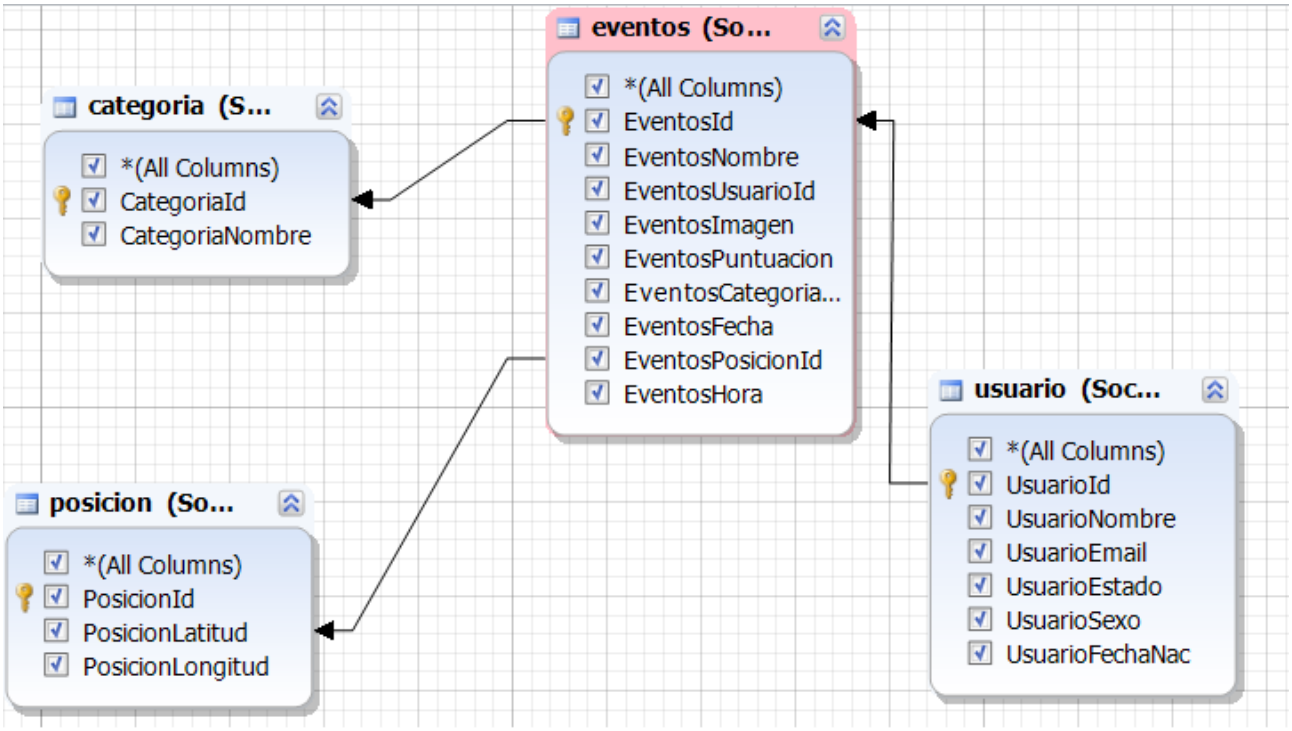
Algunas Desventajas de UML:

✓ El tiempo para obtener un sistema funcional es muy superior al que se necesita en el Diseño de Base de Datos.
✓ Se requieren múltiples ciclos de desarrollo para obtener un sistema funcional de software que responda debidamente a los requerimientos.
✓ Al terminar un ciclo, la complejidad del sistema crece al incorporarle nuevas funciones.
✓ En cada ciclo se aborda un conjunto relativamente pequeño de requerimientos, impidiendo obtener una visión global del sistema.
✓ Para los requerimientos planteados no se obtiene una solución óptima en tiempos aceptables.
✓ Los costos son elevados.

Ventajas del Diseño de Base de Datos:

✓ Se obtiene un conjunto de especificaciones que contempla correctamente los requerimientos del usuario en un tiempo mucho menor al que necesita la Metodología UML.
✓ La complejidad del sistema se encuentra en rangos aceptables.
✓ Las actividades de relevamiento, análisis, diseño, desarrollo, implementación y prueba del sistema de información, se llevan a cabo una vez que se cuenta con todos los requerimientos del sistema posibilitando tener una visión global del mismo.
✓ Para la situación problemática planteada se obtiene una solución óptima en tiempos aceptables.
✓ Los costos son regulares.

5.2 Esquema de la base de Datos:



5.3 Desarrollo

Con todo lo expuesto y fundamentado en el punto anterior, se procedió a realizar el **Desarrollo de la Aplicación** mediante un **Lenguaje de Programación Visual Orientado a Objetos,** como lo es Adobe Dreamweaver CS6, utilizando los requerimientos obtenidos al aplicar la Metodología de Diseño de Bases de Datos mediante el Modelo Entidad Relación y el Modelo Relacional.

Características fundamentales de Adobe Dreamweaver CS6 como **Lenguaje de Programación Visual Orientado a Objetos:**

- ✓ Dreamweaver es una excelente herramienta, al menos la mejor en el área de pre visualización, y cualquier empresa que quiera tener un sitio web debería tener al menos una copia del programa para tener un buen resultado. Lo que pasa es que ahora los tiempos cambian y los desarrolladores quieren que algo se vea cómo se les solicita, tienen hacerlo con código paso por paso; esa es la tendencia con HTML5 y CSS3, y por eso Dreamweaver se vuelve más un complemento para hacer algo increíble pero rápido.
- ✓ Las clases de objetos son manejadas internamente por el lenguaje.

- ✓ La interacción entre los objetos de software participantes es manejada internamente por el lenguaje. Es decir, el usuario no necesita preocuparse por modelar los objetos, clases y sus interacciones.

5.4 Programación de la Aplicación de Software

Codificación

Se hará realidad el modelo de diseño, es decir que se elaboran y adaptan los elementos gráficos y lógicos, se codifican los módulos, los programas, se definen y preparan las bases de datos para que el proyecto quede en completo funcionamiento.

En esta etapa hay que realizar pruebas exhaustivas para asegurar el perfecto funcionamiento del sistema, se prueba la integración con los sistemas internos, todo esto se hace primero en el ambiente de desarrollo.

Conexión de la Aplicación con la Base de Datos.

Código PHP utilizado para conectarse con la Base de Datos:

conectar.php

```
<?php
$db_host="localhost";
$db_usuario="root";
$db_password="";
$db_nombre="serrano";
$conexion = @mysql_connect($db_host, $db_usuario, $db_password) or die(mysql_error());
$db = @mysql_select_db($db_nombre, $conexion) or die(mysql_error());
?>
```

desconectar.php

```
<?php
@mysql_close($conexion);
?>
```

Pasos para establecer una conexión:

Se debe hacer desde Windows:

Inicio -> Panel de Control -> Herramientas Administrativas-> Orígenes de Datos (ODBC)

->DSN de Sistema -> Agregar una nueva conexión para MySQL mediante un alias.

Componentes de Adobe Dreamweaver CS6 utilizados: Formulario, Cajas de Texto, Botones, Grilla, Etiquetas.

5.5 Código fuente (Módulos más relevantes)

Index.php

```
<?php

session_start();

//verifico la sesion que deberia haber llegado desde registro
if (empty($_SESSION["facebook_id"])){

 require_once("src/facebook.php");
 require_once("src/config.php");

  $facebook = new Facebook($config);

$params = array(
  'scope' => 'read_stream, friends_likes',
  'redirect_uri' => 'http://www.socialdroidpro.com/registro.php'
);

$loginUrl = $facebook->getLoginUrl($params);



}

?>



<!DOCTYPE HTML>
<html>

<head>
  <title>SocialDroid Project 2014 - Tucum&aacute;n, Argentina - FRT-UTN Proyecto 2014</title>
  <meta name="description" content="website description" />
  <meta name="keywords" content="website keywords, website keywords" />
  <meta http-equiv="content-type" content="text/html; charset=UTF-8" />
  <link rel="stylesheet" type="text/css" href="css/style.css" />
  <!-- modernizr enables HTML5 elements and feature detects -->
  <script type="text/javascript" src="js/modernizr-1.5.min.js"></script>
</head>



<div id="main">
    <header>
      <div id="logo">
        <div id="logo_text">
          <!-- class="logo_colour", allows you to change the colour of the text -->
          <h1><a href="index.html">Eventos <span
```

```
class="logo_colour">Tucumanos</span></a></h1>
        <h2>Sucedi&oacute;... Sucede... Suceder&aacute;... </h2>
      </div>
    </div>
    <nav>
      <ul class="sf-menu" id="nav">
        <li class="selected"><a href="index.php">Inicio</a></li>

<?php if (!empty($_SESSION["facebook_id"])){         // solo muestro si alguien esta logueado
?>
        <li><a href="index.php?Accion=1" id="new">Crear Evento</a></li>
            <li><a href="index.php?Accion=2" id="mod">Modificar Evento</a></li>
        <li><a href="index.php?Accion=3" id="find">Buscar Evento</a></li>
<?php

}
?>

      <li><a href="index.php?Accion=4" id="top">Top 10</a></li>

<?php if (empty($_SESSION["facebook_id"])){

?>

<li><a href="<?php echo $loginUrl;  ?>">Login con Facebook</a></li>

<?php

}
else{

?>

<li><a href="logout.php"><?php echo "Bienvenido ".$_SESSION["name"]." "; ?> Logout</a></li>

<?php

}

?>

      </ul>
    </nav>
  </header>

<body>



  <div id="site_content">
    <ul id="images">
      <li><img src="images/1.jpg" width="600" height="300" alt="seascape_one" /></li>
      <li><img src="images/2.jpg" width="600" height="300" alt="seascape_two" /></li>
      <li><img src="images/3.jpg" width="600" height="300" alt="seascape_three" /></li>
      <li><img src="images/4.jpg" width="600" height="300" alt="seascape_four" /></li>
      <li><img src="images/5.jpg" width="600" height="300" alt="seascape_five" /></li>
      <li><img src="images/6.jpg" width="600" height="300" alt="seascape_seascape" /></li>
    </ul>
    <div id="sidebar_container">
      <div class="sidebar">
        <h3>Nuevos eventos y categorias</h3>
```

```
        <h4>Nueva categoria "Otros"</h4>
        <h5>Febrero 1ro, 2014</h5>
        <p>Eventos deportivos.<br /><a href="#">Read more</a></p>
        <h4>Clasico tucumano</h4>
        <h5>Imagenes y comentarios</h5>
        <p>Se publicaron 0 eventos en la fecha.<br /><a href="#">Read more</a></p>
      </div>
    </div>



<?php

$accion = $_GET['Accion'];

if ($accion == "1") {

//echo 1;

?>
<h3>Crear evento:</h3>
<br><br>
<form action="crearevento.php" method="post" enctype="multipart/form-data" >
Nombre del evento:
<br><br><input type="text" size="15" maxlength="30" name="nombre"><br><br>
Comentarios:
<br><br><textarea name="comentarios" rows="5" cols="51">Escribe aquí tus
comentarios</textarea><br><br>
Categoria:
<br><br><select name="categoria">
<option>Accidente</option>
<option>Manifestaciòn</option>
<option>Entretenimiento</option>
<option>Deportivo</option>
<option>Otros</option>
</select><br><br>
Imagen:
<input type=file name="file1"><br><br>
<input type="submit" value="Enviar">
</form><br>
<?php




}else
        if ($accion == "2") {

//      echo 2;

        ?>
<h3>Administrar evento:</h3>


Seleccione su Evento:
<br><br><select name="EVento">
<option>Accidente alderetes</option>
<option>Manifestaciòn en las talitas</option>
<option>Recital Illya</option>
<option>Clasico</option>
<option>Convencion</option>
</select><br><br>
```

```
Comentarios:
<br><br><textarea name="Comentarios" rows="5" cols="51">Escribe aquí tus
comentarios</textarea><br><br>
Categoria:
<br><br><select name="Categoria">
<option>Accidente</option>
<option>Manifestaciòn</option>
<option>Entretenimiento</option>
<option>Deportivo</option>
<option>Otros</option>
</select><br><br>
Imagen:
<br><br><form method="post" enctype="multipart/form-data"><br>
<input type=file size=60 name="file1"><br><br>
</form><br>
<input type="submit" value="Enviar">
<input type="reset" value="Limpiar">
<?php

        }else
                if ($accion == "3"){

//              echo 3;
?>
<h3>Buscar evento:</h3>
<p> </p>

Categoria:
<br><br><select name="Categoria">
<option>Accidente</option>
<option>Manifestaciòn</option>
<option>Entretenimiento</option>
<option>Deportes</option>
<option>Otros</option>
</select><br><br>

Evento:
<br><br><select name="EVento">
<option>Accidente alderetes</option>
<option>Manifestaciòn en las talitas</option>
<option>Recital Illya</option>
<option>Clasico</option>
<option>Convencion</option>
</select><br><br>
Comentarios:
<br><br><textarea readonly name="Comentarios" rows="5" cols="51">Escribe aquí tus
comentarios</textarea><br><br>

Imagen:
<br><br>
<input type="image" name="boton" src="images/1.jpg" align="middle">
<br><br>
<input type="submit" value="Publicar en Face">
<input type="reset" value=Twittear">
<?php
                }else
                if ($accion == "4"){

//              echo 3;
?>
<h3>Top 10:</h3>
<p> </p>
```

```
Categoria:
<br><br><select name="Categoria">
<option>Accidente</option>
<option>Manifestaciòn</option>
<option>Entretenimiento</option>
<option>Deportes</option>
<option>Otros</option>
</select><br><br>


Imagen:
<br><br>
<input type="image" name="boton" src="images/1.jpg" align="middle">
<br><br>
<input type="submit" value="Publicar en Face">
<input type="reset" value=Twittear">
<?php
                    }



?>



   </div>
   <footer>
    <p>Alumnos: &copy; Cristian Roberto Mene Leg: 23719 - Juan Villegas Leg: 20630 </p>
   </footer>
  </div>
  <p> </p>
  <!-- javascript at the bottom for fast page loading -->
  <script type="text/javascript" src="js/jquery.js"></script>
  <script type="text/javascript" src="js/jquery.easing-sooper.js"></script>
  <script type="text/javascript" src="js/jquery.sooperfish.js"></script>
  <script type="text/javascript" src="js/jquery.kwicks-1.5.1.js"></script>
  <script type="text/javascript">
   $(document).ready(function() {
    $('#images').kwicks({
     max : 600,
     spacing : 2
    });
    $('ul.sf-menu').sooperfish();
   });
  </script>
</body>
</html>
```

Conexión.php

```
<?php
$db_host="localhost";
$db_usuario="root";
$db_password="tecno";
$db_nombre="socialdroid";
$conexion = @mysql_connect($db_host, $db_usuario, $db_password) or die(mysql_error());
$db = @mysql_select_db($db_nombre, $conexion) or die(mysql_error());
?>
```

Config.php

```
<?php

$config = array();
$config['appId']  = '238459269660297';
$config['secret'] = '67e7b44fef0221f42a3d5486261fc9e0';

$server = 'localhost';
$user = 'root';
$pass = 'tecno';
$db = 'socialdroid';

?>
```

Contacto.php

```
<!DOCTYPE HTML>
<html>

<head>
  <title>CSS3_seascape_two</title>
  <meta name="description" content="website description" />
  <meta name="keywords" content="website keywords, website keywords" />
  <meta http-equiv="content-type" content="text/html; charset=UTF-8" />
  <link rel="stylesheet" type="text/css" href="css/style.css" />
  <!-- modernizr enables HTML5 elements and feature detects -->
  <script type="text/javascript" src="js/modernizr-1.5.min.js"></script>
</head>

<body>
  <div id="main">
    <header>
      <div id="logo">
        <div id="logo_text">
          <!-- class="logo_colour", allows you to change the colour of the text -->
          <h1><a href="index.html">CSS3<span
class="logo_colour">_seascape_two</span></a></h1>
          <h2>Simple. Contemporary. Website Template.</h2>
        </div>
      </div>
      <nav>
        <ul class="sf-menu" id="nav">
          <li><a href="index.html">Home</a></li>
          <li><a href="examples.html">Examples</a></li>
          <li><a href="page.html">A Page</a></li>
          <li><a href="another_page.html">Another Page</a></li>
          <li><a href="#">Example Drop Down</a>
            <ul>
              <li><a href="#">Drop Down One</a></li>
              <li><a href="#">Drop Down Two</a>
                <ul>
                  <li><a href="#">Sub Drop Down One</a></li>
                  <li><a href="#">Sub Drop Down Two</a></li>
                  <li><a href="#">Sub Drop Down Three</a></li>
                  <li><a href="#">Sub Drop Down Four</a></li>
                  <li><a href="#">Sub Drop Down Five</a></li>
                </ul>
              </li>
```

```
            <li><a href="#">Drop Down Three</a></li>
            <li><a href="#">Drop Down Four</a></li>
            <li><a href="#">Drop Down Five</a></li>
          </ul>
        </li>
        <li class="selected"><a href="contact.php">Contact Us</a></li>
      </ul>
    </nav>
  </header>
  <div id="site_content">
    <ul id="images">
      <li><img src="images/1.jpg" width="600" height="300" alt="seascape_one" /></li>
      <li><img src="images/2.jpg" width="600" height="300" alt="seascape_two" /></li>
      <li><img src="images/3.jpg" width="600" height="300" alt="seascape_three" /></li>
      <li><img src="images/4.jpg" width="600" height="300" alt="seascape_four" /></li>
      <li><img src="images/5.jpg" width="600" height="300" alt="seascape_five" /></li>
      <li><img src="images/6.jpg" width="600" height="300" alt="seascape_seascape" /></li>
    </ul>
    <div id="sidebar_container">
      <div class="sidebar">
        <h3>Latest News</h3>
        <h4>New Website Launched</h4>
        <h5>January 1st, 2012</h5>
        <p>2012 sees the redesign of our website. Take a look around and let us know what you
think.<br /><a href="#">Read more</a></p>
        <h4>20% Discount</h4>
        <h5>March 1st, 2012</h5>
        <p>We are offering a 20% discount to all new customers.<br /><a href="#">Read
more</a></p>
      </div>
    </div>
    <div class="content">
      <h1>Contact Us</h1>
      <p>Say hello, using this contact form.</p>
      <?php
        // This PHP Contact Form is offered &amp;quot;as is&amp;quot; without warranty of any kind, either
expressed or implied.
        // David Carter at www.css3templates.co.uk shall not be liable for any loss or damage arising
from, or in any way
        // connected with, your use of, or inability to use, the website templates (even where David
Carter has been advised
        // of the possibility of such loss or damage). This includes, without limitation, any damage for
loss of profits,
        // loss of information, or any other monetary loss.

        // Set-up these 3 parameters
        // 1. Enter the email address you would like the enquiry sent to
        // 2. Enter the subject of the email you will receive, when someone contacts you
        // 3. Enter the text that you would like the user to see once they submit the contact form
        $to = 'enter email address here';
        $subject = 'Enquiry from the website';
        $contact_submitted = 'Your message has been sent.';

        // Do not amend anything below here, unless you know PHP
        function email_is_valid($email) {
          return preg_match('/^[A-Z0-9._%+-]+@[A-Z0-9.-]+\.[A-Z]{2,4}$/i',$email);
        }
        if (!email_is_valid($to)) {
          echo '<p style="color: red;">You must set-up a valid (to) email address before this contact
page will work.</p>';
        }
        if (isset($_POST['contact_submitted'])) {
```

```
          $return = "\r";
          $youremail = trim(htmlspecialchars($_POST['your_email']));
          $yourname = stripslashes(strip_tags($_POST['your_name']));
          $yourmessage = stripslashes(strip_tags($_POST['your_message']));
          $contact_name = "Name: ".$yourname;
          $message_text = "Message: ".$yourmessage;
          $user_answer = trim(htmlspecialchars($_POST['user_answer']));
          $answer = trim(htmlspecialchars($_POST['answer']));
          $message = $contact_name . $return . $message_text;
          $headers = "From: ".$youremail;
          if (email_is_valid($youremail) && !eregi("\r",$youremail) && !eregi("\n",$youremail) &&
$yourname != "" && $yourmessage != "" && substr(md5($user_answer),5,10) === $answer) {
            mail($to,$subject,$message,$headers);
            $yourname = '';
            $youremail = '';
            $yourmessage = '';
            echo '<p style="color: blue;">'.$contact_submitted.'</p>';
          }
          else echo '<p style="color: red;">Please enter your name, a valid email address, your
message and the answer to the simple maths question before sending your message.</p>';
        }
        $number_1 = rand(1, 9);
        $number_2 = rand(1, 9);
        $answer = substr(md5($number_1+$number_2),5,10);
       ?>
       <form id="contact" action="contact.php" method="post">
        <div class="form_settings">
          <p><span>Name</span><input class="contact" type="text" name="your_name"
value="<?php echo $yourname; ?>" /></p>
          <p><span>Email Address</span><input class="contact" type="text" name="your_email"
value="<?php echo $youremail; ?>" /></p>
          <p><span>Message</span><textarea class="contact textarea" rows="5" cols="50"
name="your_message"><?php echo $yourmessage; ?></textarea></p>
          <p style="line-height: 1.7em;">To help prevent spam, please enter the answer to this
question:</p>
          <p><span><?php echo $number_1; ?> + <?php echo $number_2; ?> = ?</span><input
type="text" name="user_answer" /><input type="hidden" name="answer" value="<?php echo
$answer; ?>" /></p>
          <p style="padding-top: 15px"><span> </span><input class="submit" type="submit"
name="contact_submitted" value="send" /></p>
        </div>
       </form>
     </div>
   </div>
   <footer>
     <p>Copyright &copy; CSS3_seascape_two | <a href="http://www.css3templates.co.uk">design
from css3templates.co.uk</a></p>
   </footer>
  </div>
  <p> </p>
  <!-- javascript at the bottom for fast page loading -->
  <script type="text/javascript" src="js/jquery.js"></script>
  <script type="text/javascript" src="js/jquery.easing-sooper.js"></script>
  <script type="text/javascript" src="js/jquery.sooperfish.js"></script>
  <script type="text/javascript" src="js/jquery.kwicks-1.5.1.js"></script>
  <script type="text/javascript">
   $(document).ready(function() {
     $('#images').kwicks({
       max : 600,
       spacing : 2
     });
     $('ul.sf-menu').sooperfish();
```

```
   });
  </script>
</body>
</html>
```

Crear.php

```
<?php
print $val=$_GET["val"];
include "conexion.php";
$sql="insert into usuarios (UsuarioNombre, UsuarioFechaNac, UsuarioSexo, UsuarioEmail) values
('$nombre', '$fechanac','$sexo','$mail')";
$result=mysql_query($sql, $conexion);
include "desconectar.php";
?>
```

Crearevento.php

```
<?php
require_once("src/config.php");

$nombreevento = htmlspecialchars($_POST["nombre"]);
echo $nombreevento;
$categoriaevento = htmlspecialchars($_POST["categoria"]);
echo $categoriaevento;


$imagen=$_FILES["file1"];

$nombre=$_FILES["file1"]["name"];

$archivo="images/".$nombre;
echo $archivo;

while(file_exists($archivo))

{

mt_srand (time());

$numero = mt_rand(0,1000);

$aux=explode(".",$nombre);

$tamano = sizeof($aux);

$extension=$aux[$tamano-1];//coge la extension de la imagen

$pos=0;

$nombre="";

while($pos<$tamano-1)//excluye la extension para crear el nombre de la imagen

{
```

```
$nombre.=$aux[$pos];

$pos=$pos+1;

}

$nombre=$nombre.$numero."".$extension;
$archivo="images/".$nombre;

}

$nombre=$nombre;
//echo $nombre;
if(move_uploaded_file($imagen["tmp_name"], "images/".$nombre))

{

echo "Subido correctamente";

}else{
//print_r(error_get_last());
echo "Se ha producido un error";

}



$mysqli = new mysqli($server, $user, $pass, $db);

//$categoriaevento
$usuarioid = 1;
$imagen = $nombre;
$posicionid = "20-30";

$categoriaid = $mysqli->query("select top 1 CategoriaId from categoria where CategoriaNombre = '". $categoriaevento. "'");
echo $categoriaid;

//$sql="insert into eventos
//(EventosNombre, EventosUsuarioId, EventosImagen, EventosPuntuacion,EventosCategoriaId,EventosFecha,EventosPosicionId,EventosHora)
//values
//('$nombreevento', '$usuarioid','$imagen',0,'$categoriaid',cast(NOW() as date), '$posicionid', cast(NOW() as time))";

//echo $sql;

//$result=$query = $mysqli->query($sql);


print_r(error_get_last());

?>
```

Facebook.php

```
<?php
/**
 * Copyright 2011 Facebook, Inc.
 *
 * Licensed under the Apache License, Version 2.0 (the "License"); you may
 * not use this file except in compliance with the License. You may obtain
 * a copy of the License at
 *
 *     http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS, WITHOUT
 * WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the
 * License for the specific language governing permissions and limitations
 * under the License.
 */

require_once "base_facebook.php";

/**
 * Extends the BaseFacebook class with the intent of using
 * PHP sessions to store user ids and access tokens.
 */
class Facebook extends BaseFacebook
{
  /**
   * Cookie prefix
   */
  const FBSS_COOKIE_NAME = 'fbss';

  /**
   * We can set this to a high number because the main session
   * expiration will trump this.
   */
  const FBSS_COOKIE_EXPIRE = 31556926; // 1 year

  /**
   * Stores the shared session ID if one is set.
   *
   * @var string
   */
  protected $sharedSessionID;

  /**
   * Identical to the parent constructor, except that
   * we start a PHP session to store the user ID and
   * access token if during the course of execution
   * we discover them.
   *
   * @param array $config the application configuration. Additionally
   * accepts "sharedSession" as a boolean to turn on a secondary
   * cookie for environments with a shared session (that is, your app
   * shares the domain with other apps).
   *
   * @see BaseFacebook::__construct
   */
  public function __construct($config) {
    if (!session_id()) {
      session_start();
```

```
    }
    parent::__construct($config);
    if (!empty($config['sharedSession'])) {
      $this->initSharedSession();

      // re-load the persisted state, since parent
      // attempted to read out of non-shared cookie
      $state = $this->getPersistentData('state');
      if (!empty($state)) {
        $this->state = $state;
      } else {
        $this->state = null;
      }

    }
  }

  /**
   * Supported keys for persistent data
   *
   * @var array
   */
  protected static $kSupportedKeys =
    array('state', 'code', 'access_token', 'user_id');

  /**
   * Initiates Shared Session
   */
  protected function initSharedSession() {
    $cookie_name = $this->getSharedSessionCookieName();
    if (isset($_COOKIE[$cookie_name])) {
      $data = $this->parseSignedRequest($_COOKIE[$cookie_name]);
      if ($data && !empty($data['domain']) &&
          self::isAllowedDomain($this->getHttpHost(), $data['domain'])) {
        // good case
        $this->sharedSessionID = $data['id'];
        return;
      }
      // ignoring potentially unreachable data
    }
    // evil/corrupt/missing case
    $base_domain = $this->getBaseDomain();
    $this->sharedSessionID = md5(uniqid(mt_rand(), true));
    $cookie_value = $this->makeSignedRequest(
      array(
        'domain' => $base_domain,
        'id' => $this->sharedSessionID,
      )
    );
    $_COOKIE[$cookie_name] = $cookie_value;
    if (!headers_sent()) {
      $expire = time() + self::FBSS_COOKIE_EXPIRE;
      setcookie($cookie_name, $cookie_value, $expire, '/', '.'.$base_domain);
    } else {
      // @codeCoverageIgnoreStart
      self::errorLog(
        'Shared session ID cookie could not be set! You must ensure you '.
        'create the Facebook instance before headers have been sent. This '.
        'will cause authentication issues after the first request.'
      );
      // @codeCoverageIgnoreEnd
    }
```

```
}

/**
 * Provides the implementations of the inherited abstract
 * methods. The implementation uses PHP sessions to maintain
 * a store for authorization codes, user ids, CSRF states, and
 * access tokens.
 */

/**
 * {@inheritdoc}
 *
 * @see BaseFacebook::setPersistentData()
 */
protected function setPersistentData($key, $value) {
  if (!in_array($key, self::$kSupportedKeys)) {
    self::errorLog('Unsupported key passed to setPersistentData.');
    return;
  }

  $session_var_name = $this->constructSessionVariableName($key);
  $_SESSION[$session_var_name] = $value;
}

/**
 * {@inheritdoc}
 *
 * @see BaseFacebook::getPersistentData()
 */
protected function getPersistentData($key, $default = false) {
  if (!in_array($key, self::$kSupportedKeys)) {
    self::errorLog('Unsupported key passed to getPersistentData.');
    return $default;
  }

  $session_var_name = $this->constructSessionVariableName($key);
  return isset($_SESSION[$session_var_name]) ?
    $_SESSION[$session_var_name] : $default;
}

/**
 * {@inheritdoc}
 *
 * @see BaseFacebook::clearPersistentData()
 */
protected function clearPersistentData($key) {
  if (!in_array($key, self::$kSupportedKeys)) {
    self::errorLog('Unsupported key passed to clearPersistentData.');
    return;
  }

  $session_var_name = $this->constructSessionVariableName($key);
  if (isset($_SESSION[$session_var_name])) {
    unset($_SESSION[$session_var_name]);
  }
}

/**
 * {@inheritdoc}
 *
 * @see BaseFacebook::clearAllPersistentData()
 */
```

```
  protected function clearAllPersistentData() {
    foreach (self::$kSupportedKeys as $key) {
      $this->clearPersistentData($key);
    }
    if ($this->sharedSessionID) {
      $this->deleteSharedSessionCookie();
    }
  }

  /**
   * Deletes Shared session cookie
   */
  protected function deleteSharedSessionCookie() {
    $cookie_name = $this->getSharedSessionCookieName();
    unset($_COOKIE[$cookie_name]);
    $base_domain = $this->getBaseDomain();
    setcookie($cookie_name, '', 1, '/', '.'.$base_domain);
  }

  /**
   * Returns the Shared session cookie name
   *
   * @return string The Shared session cookie name
   */
  protected function getSharedSessionCookieName() {
    return self::FBSS_COOKIE_NAME . '_' . $this->getAppId();
  }

  /**
   * Constructs and returns the name of the session key.
   *
   * @see setPersistentData()
   * @param string $key The key for which the session variable name to construct.
   *
   * @return string The name of the session key.
   */
  protected function constructSessionVariableName($key) {
    $parts = array('fb', $this->getAppId(), $key);
    if ($this->sharedSessionID) {
      array_unshift($parts, $this->sharedSessionID);
    }
    return implode('_', $parts);
  }
}
```

Generado.php

```
<!DOCTYPE HTML>
<html>
<head>
  <title>FRT-UTN Proyecto 2014</title>
  <meta name="description" content="website description" />
  <meta name="keywords" content="website keywords, website keywords" />
  <meta http-equiv="content-type" content="text/html; charset=UTF-8" />
  <link rel="stylesheet" type="text/css" href="css/style.css" />
  <!-- modernizr enables HTML5 elements and feature detects -->
  <script type="text/javascript" src="js/modernizr-1.5.min.js"></script>
</head>";

</php>
```

Login.php

```
<div id="fb-root"></div>
<script>
  window.fbAsyncInit = function() {
  FB.init({
    appId      : '238459269660297',
    status     : true, // check login status
    cookie     : true, // enable cookies to allow the server to access the session
    xfbml      : true  // parse XFBML
  });

  // Here we subscribe to the auth.authResponseChange JavaScript event. This event is fired
  // for any authentication related change, such as login, logout or session refresh. This means that
  // whenever someone who was previously logged out tries to log in again, the correct case below
  // will be handled.
  FB.Event.subscribe('auth.authResponseChange', function(response) {
    // Here we specify what we do with the response anytime this event occurs.
    if (response.status === 'connected') {
      // The response object is returned with a status field that lets the app know the current
      // login status of the person. In this case, we're handling the situation where they
      // have logged in to the app.
      testAPI();
    } else if (response.status === 'not_authorized') {
      // In this case, the person is logged into Facebook, but not into the app, so we call
      // FB.login() to prompt them to do so.
      // In real-life usage, you wouldn't want to immediately prompt someone to login
      // like this, for two reasons:
      // (1) JavaScript created popup windows are blocked by most browsers unless they
      // result from direct interaction from people using the app (such as a mouse click)
      // (2) it is a bad experience to be continually prompted to login upon page load.
      FB.login();
    } else {
      // In this case, the person is not logged into Facebook, so we call the login()
      // function to prompt them to do so. Note that at this stage there is no indication
      // of whether they are logged into the app. If they aren't then they'll see the Login
      // dialog right after they log in to Facebook.
      // The same caveats as above apply to the FB.login() call here.
      FB.login();
    }
  });
  };

  // Load the SDK asynchronously
  (function(d){
   var js, id = 'facebook-jssdk', ref = d.getElementsByTagName('script')[0];
   if (d.getElementById(id)) {return;}
   js = d.createElement('script'); js.id = id; js.async = true;
   js.src = "//connect.facebook.net/en_US/all.js";
   ref.parentNode.insertBefore(js, ref);
  }(document));

  // Here we run a very simple test of the Graph API after login is successful.
  // This testAPI() function is only called in those cases.
  function testAPI() {
    console.log('Welcome!  Fetching your information.... ');
    FB.api('/me', function(response) {
      console.log('Good to see you, ' + response.name + '.');
    });
```

```
  }
</script>

<!--
  Below we include the Login Button social plugin. This button uses the JavaScript SDK to
  present a graphical Login button that triggers the FB.login() function when clicked. -->


<fb:login-button show-faces="true" width="200" max-rows="1"></fb:login-button>
```

Logout.php

```
<?php
session_start();
session_destroy();

header("Location:index.php");

?>
```

Registro.php

```
<?php
session_start();
require_once("src/facebook.php");
require_once("src/config.php");


$facebook = new Facebook($config);
$facebook_id = $facebook->getUser();



if($facebook_id!=0){
//echo "esta conectado"

//conectar a la DB

$mysqli = new mysqli($server, $user, $pass, $db);

$query = $mysqli->query("select * from usuario where UsuarioFacebookId = '$facebook_id'");

if ($query->num_rows<>0){

//ya existe

$datos = $facebook->api("/me");

//buscamos el user en mysql

$datos=$query->fetch_array(MYSQLI_ASSOC);

$_SESSION["facebook_id"] = $facebook_id;
$_SESSION["name"] = $datos["UsuarioNombre"];
$_SESSION["email"] = $datos["UsuarioEmail"];

}else {

//obtenemos los datos del usuario
```

```
$datos = $facebook -> api("/me");

$Nombre   = $datos["name"];
$Email    = $datos["email"];
$Sexo     = $datos["genrer"];
$FechaNac = $datos["birthday"];


//no existe
$insert_user = $mysqli->query("INSERT INTO usuario(
  UsuarioNombre,
  UsuarioEmail,
  UsuarioEstado,
  UsuarioSexo,
  UsuarioFechaNac,
  UsuarioFacebookId
  )
  values(
'$Nombre','$Email','1','$Sexo','$FechaNac','$facebook_id'
  )
");//fin de insert

$_SESSION["facebook_id"] = $facebook_id;
$_SESSION["name"] = $Nombre;
$_SESSION["email"] = $Email;

}



}
//else {echo "no esta conectado"}

header("location:index.php");

?>
```

5.6 Depuración y prueba de la Aplicación

Se habilitará el proyecto para que los verdaderos usuarios comiencen a servirse del sistema.

El proyecto se encuentra en pleno funcionamiento con usuarios verdaderos accediendo y con los administradores realizando las tareas periódicas de mantenimiento.

Evolución

Como todo sistema, para conseguir su objetivo, el proyecto seguirá evolucionando adaptándose a su medio ambiente. Aquí deben definirse las características de tal evolución.

Para esto debe analizarse continuamente su funcionamiento, considerando los resultados planificados, los realmente obtenidos y el comportamiento del usuario, para realizar los ajustes cuando sea apropiado.

Depuración del Sistema:

Actividades realizadas:

Análisis de los mensajes mostrados por el compilador.

Detección de errores durante la carga de datos.

Seguimiento realizado por cada un de los módulos del sistema.

Detección de omisiones e incorporación de las mismas al sistema.

Prueba del Sistema:

Actividades realizadas:

Se hizo funcionar el sistema mediante la carga de datos.

Prueba de la conexión de la aplicación con el motor de Base de Datos.

Ingreso de datos no permitidos.

5.7 Elaboración del Manual del usuario

Actividades realizadas:

Determinación del contenido.
Construcción del mismo.

Temas abordados

- Acerca del sistema
- Instalación del sistema
- Módulos incorporados al sistema
- Funciones y utilidad de los módulos
- Acerca de los componentes visuales del Lenguaje de Programación usado
- Uso de la ayuda del sistema
- Acerca del ingreso de los datos
- Uso de los eventos para el bien público
- Manejo de los reportes y utilización de los mismos

5.8 Análisis de costos

Pasaremos a realizar los cálculos necesarios para determinar el costo que implica el desarrollo de la aplicación de software “SocialDroid”.

Se determinan solamente los costos variables relacionados directamente con “la mano de obra” necesaria para producir este software, no se calcula la inversión necesaria para montar un gabinete de desarrollo de software ni los costos fijos que inciden, dando por supuesto que disponemos de los equipos necesarios para desarrollar la aplicación.

Las cifras que se obtienen son solamente estimativas derivadas de la base del tiempo promedio que se necesita para desarrollar un proyecto de software de estas características y los valores tomados para realizar los cálculos son los que ofrece el mercado laboral actual de la provincia de Tucumán.

Cálculos

Tiempo promedio estimado para finalizar el proyecto:

2 meses. 20 días hábiles por mes x 2 = 40 días hábiles

Personal de sistemas necesario:

2 Analistas Universitarios de Sistemas

Tiempo que se necesita el personal:

2 Analistas Universitarios de Sistemas durante 2 meses

Costo de la mano de obra para el Proyecto:

Analista Universitario de Sistemas – 2 meses = 40 días hábiles

1 día laboral = 8 horas

8 [horas/día] * 40 [días] = **320 [horas]**

1 hora laboral = **$ 62,70**

Dato obtenido desde http://www.sup.org.ar/salarios/salarios-agencias.html

Costo total mano de obra Analista = 62,70 [$/hora] * 320 [horas]

Costo total mano de obra de cada Analista = 20064,00 [$]

Costo Total de la mano de obra para el Proyecto:

Costo total = Costo total mano de obra Analista * 2

Costo total = 40128,00 [$]

Costo total = 40128,00 [$]

Flujo de caja.

Egresos x período mensual	**1**	**2**
Analistas	20064,00 [$]	20064,00 [$]
Total	20064,00 [$]	20064,00 [$]

Punto de equilibrio

Por tratarse de un software gratuito al que pueden acceder desde cualquier lugar, el punto de equilibrio estaría determinado por el tiempo que tardarían las contribuciones de los distintos auspiciantes publicitarios en nuestro sistema en cubrir el Flujo de egreso de caja de $ 40128,00; el cual no se puede estimar con precisión.

5.9 Conclusiones

Se desarrolló un Paquete de Software a medida, para gestionar la información que se maneja en los eventos que se publican a través de las redes sociales de mayor envergadura de la actualidad, como son Facebook y Twitter, combinando la información suministrada por los millones de usuarios de manera eficiente y optimizándola a las necesidades requeridas por la sociedad, cubriendo con todas las expectativas exigidas, siendo la primer aplicación que combine las redes sociales de mayor auge.

Esta aplicación brinda una solución informática gratuita, práctica, eficaz, confiable, y segura, permitiendo a los usuarios de la misma obtener el máximo beneficio, más precisamente contribuyendo con las entidades encargadas de actuar rápidamente cuando ocurre un siniestro. Es gratuita porque se puede acceder a la misma desde cualquier sitio del mundo. Práctica, porque se puede ejecutar tanto en una computadora, una tablet ó incluso desde un teléfono celular, con acceso a Internet, teniendo acceso a la información de forma instantánea. Segura, generará en forma automática copias de respaldo de los eventos ocurridos.

Respecto de las Metodologías de Modelado utilizadas, en principio modelamos el sistema mediante la Metodología O.O. con UML de Larman, en cierto punto debido a situaciones problemáticas que se presentaron, se realizó el modelado del sistema a través del Modelo Entidad Relación y el Modelo Relacional, con herramientas tales como ER-Studio 7.1 y dbForge Studio for MySQL, aprovechando las características más sobresalientes de estas herramientas, como la Reutilización de Componentes presentada por el Lenguaje de Programación O.O. utilizado, Adobe Dreamweaver CS6 y Php en este caso, armonizando todo esto con la potencia de un Motor de Base de Datos como lo es MySql.

5.10 Bibliografía

Fundamentos de Bases de Datos. Rovarini, Pablo y De la Vega, Herminia

Procesamiento de Bases de Datos. Kroenke, David

UML y Patrones. Larman, Craig

Sistemas Administrativos. Magdalena, Fernando.

Análisis y Diseño de Sistemas. Kendall, Julie y Kendall, Kenneth

“Construyendo software de alta calidad”
http://www.elguille.info/colabora/NET2005/Percynet_ConstruyendoSoftCalidad.htm