Nuevento

Estudiante: **Julio Agüero Sánchez**
Curso: **Aplicaciones Web**
Profesor: **Andréi Fuentes**
Tecnológico de Costa Rica, 2012

Contenido

Descripción del problema

Los estudiantes siempre tienen muchas cosas en las que pensar, por lo que se convierte en una tarea difícil recordarse de todas ellas. Por lo tanto se tiene la necesidad de poder tener un sitio centralizado en el que se puedan crear, mantener y revisar eventos relacionados con la universidad. Sea este una charla académica, una actividad social o cultural.

Agregado a esto, debe poder ser accedido por cualquier estudiante con un dispositivo con conexión a la Internet. Entonces se debe permitir registrar a usuarios para poder llevar un control de sus acciones y poder relacionar los eventos con un usuario en específico.

Esto hará que se pueda mejorar la asistencia a los distintos eventos de interés de la comunidad estudiantil e institucional, además de mantener a los estudiantes informados los distintos acontecimientos que podrían interesarles.

Como solución se propone la programación de una aplicación web en Ruby On Rails. En esta aplicación se cubrirán los requerimientos previamente acordados con el cliente, brindando así una plataforma tecnológica para el uso cotidiano de los estudiantes universitarios.

Diseño del programa

Descripción breve de los elementos principales

Eventos

Son el elemento primordial de la aplicación. Los usuarios podrán crear sus propios eventos con la información correspondiente a cada uno de ellos. Y cada usuario registrado podrá ver la información de los otros eventos existentes.

Categorías

Cada evento tiene su propia categoría. Se crean categorías según sea la necesidad del usuario.

Decisiones de diseño

Como se mencionó anteriormente, se utiliza el framework Rails, por lo que se sigue un patrón arquitectónico de Model View Controller. Esto principalmente para aprovechar la agilidad de desarrollo que brinda el framework. Además por la familiaridad con el patrón, lo cual permite un mejor desarrollo con mayor calidad.

También, será una aplicación web, por lo que se puede acceder con sólo poseer un browser, por lo que es *cross-platform*, una ventaja en la actualidad por la existencia de tantos sistemas operativos.

Se hará *deployment* de la aplicación usando el servicio de Heroku, de esta manera se tendrá una plataforma que será accesible desde cualquier parte del mundo con una conexión a la web.

Se utilizará PostgreSQL como base de datos y Ubuntu 11.04 como sistema operativo para el desarrollo.

Se hizo uso de dos APIs: Para la autenticación de usuarios, se usó el API que provee Twitter y para poder desplegar mapas y guardar información de posicionamiento global de los eventos, se echó mano al API de google maps.

Rails provee herramientas para el desarrollo rápido como el scaffold, por lo que se decidió utilizar esta herramienta para aprovechar las ventajas que ofrece. Por lo que nombres de convención fueron manejados por el framework, lo que facilita el trabajo aún más.

Para lograr que ciertas autorizaciones fueran efectivas se tuvo que crear cierto tipo de código embebido de decisión en las vistas para mostrar la información necesaria solamente.

Para esta versión beta, se decidió utilizar hojas de estilo CSS para brindar una interfaz de usuario más refrescante y atractiva, siempre buscando la usabilidad y sencillez.

Diagrama de clases

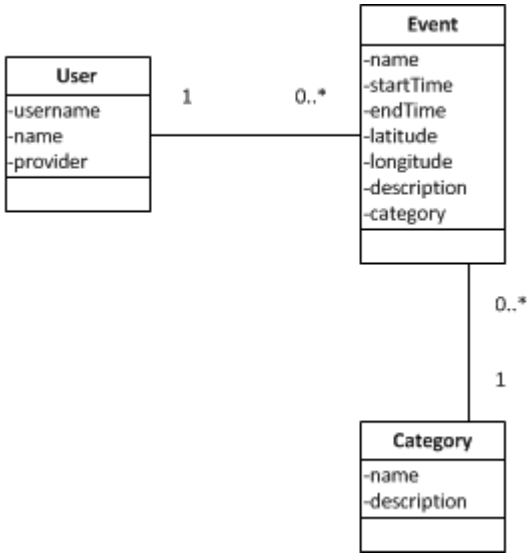
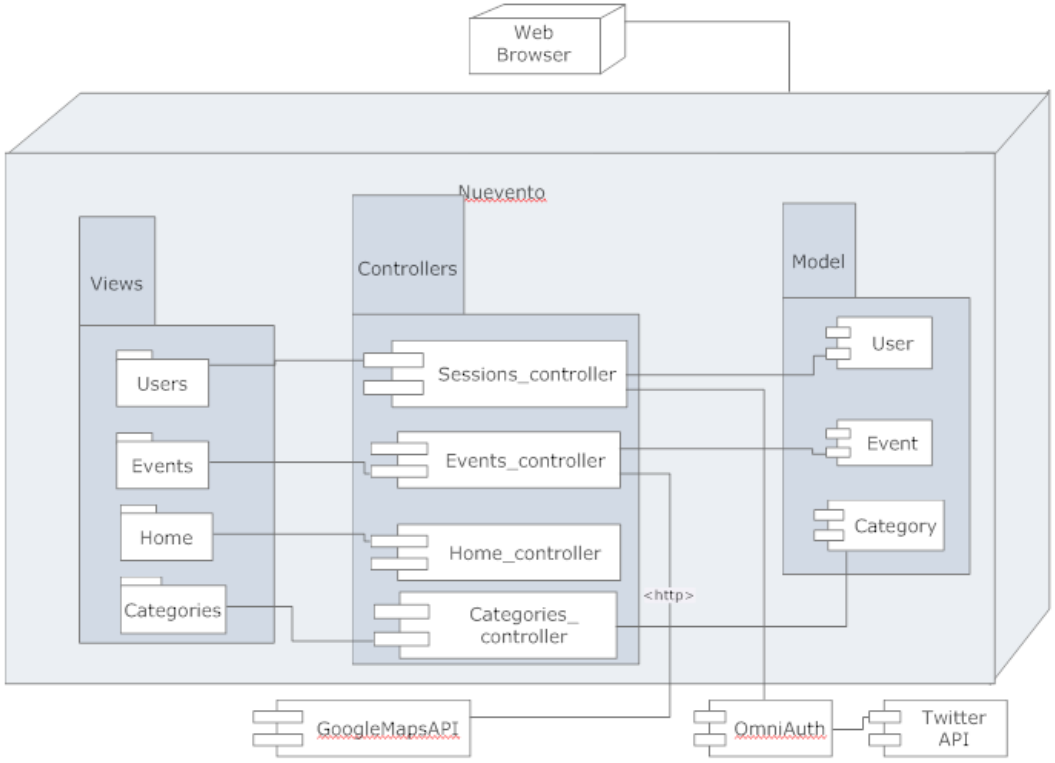


Diagrama de arquitectura

Se detallan los componentes principales y sus relaciones:



Uso de Rails

En general es un Framework que ayuda al desarrollo de aplicaciones al elevar el nivel de abstracción para el usuario, haciendo que sea más sencilla la programación. *Ruby On Rails* "enfatiza el uso de patrones y principios de ingeniería de software bien conocidos, como Active Record pattern, Convention over Configuration, 'Don't Repeat Yourself' y Model-View-Controller" (Wikipedia, 2012)

Manejo de Controladores

Como se sabe, un controlador es el componente de Rails que responde a las peticiones externas de el servidor web a la aplicación y determina cual es archivo que se debe *renderizar*. (Wikipedia, 2012)

Para esta aplicación se tienen cuatro controladores: categories_controller, events_controller, sessions_controller y home_controller. El primero maneja las solicitudes sobre el objeto categorias, el segundo se encarga de todas las solicitudes a los eventos, el tercero maneja el ingreso (log in) y la salida (log out) de los usuarios a la aplicación y el último funciona para poder mostrar la página de bienvenida a la aplicación.

A los controladores se les añadieron tanto métodos como también se modificó su funcionalidad existente para poder cumplir con los requerimientos del programa.

Manejo de Vistas

Para el formato de las vistas se usó erb, se hizo tanto una versión en HTML como en texto plano para cada uno de los métodos del CRUD de Category como también de Events. Añadido a esto, se crearon ciertas vistas como la de bienvenida al usuario y la vista por categorías de los eventos.

Se tuvo que insertar código de ruby dentro de las vistas para poder tomar ciertas decisiones de formateo de las referencias bibliográficas y decidir si renderizar algún hipervínculo u otro.

También, en las vistas se lleva a cabo la validación por parte del cliente de los datos ingresados en los distintos formularios de la aplicación.

Manejo de Rutas

Para poder agregar una acción a un controlador se tiene que asegurar que esta esté incluida en el archivo de ruteo de la aplicación, de lo contrario no se podrá utilizar. Se recomienda el uso de rutas RESTful, "que incluye acciones como: create, new, edit, update, destroy, show y index" unicamente que éstas son ruteadas automáticamente por convención.

REST o REpresentational State Transfer es "un estilo de arquitectura de software para sistemas distribuídos." (Wikipedia, 2012). Por esto es que se hace uso de todas las palabras clave para peticiones HTTP: GET, POST, PUT, DELETE, entre otras.

Se tuvo que modificar el archivo de rutas para métodos como log in, log out y categorize.

Manejo de Modelos

Se crearon tres modelos: Event, Category y User. Estos fueron creados por medio de la herramienta scaffold. Y luego se le agregó la relación belongs_to :user y belongs_to:category en el modelo Event, para indicar que un evento pertenece a un usuario y tiene una categoría de esa manera se puede acceder a la información de usuario y de categoría desde Event.

Se usó validación a nivel de modelos, para mantener a los controladores "delgados". Esta es una práctica recomendada.

Hay que considerar la nominación attr_accessible y lo importante que es colocar a cualquier atributo que sea ingresado por el usuario bajo esta categoría.

Archivos de configuración

Para poder hacer uso de postgreSQL se tuvo que modificar el archivo database.yml e ingresar las configuraciones necesarias para que la aplicación pudiera acceder a la base de datos creada para este propósito, fuera la de pruebas o desarrollo.

También, se tiene el archivo omniauth.rb, en el cual se configura correctamente el componente para poder autenticarse por medio de Twitter, lo que se cambió en este archivo es que se indica el Consumer_Secret y Consumer_Token proveídos por el API de Twitter.

Análisis de resultado

Aspecto	Estado
Gestión de Usuarios	Implementado con éxito
Autorización y Autenticación	Implementado con éxito
Gestión de Eventos	Implementado con éxito
Gestión de Categorías	Implementado con éxito
Deployment a Heroku	Implementado con éxito
Uso de Twitter API para autenticación	Implementado con éxito
Uso de Google Maps API	Implementado con éxito
Validaciones de parte del cliente	Implementado con éxito
Uso del CSS	Implementado con éxito
Documentación interna	Implementado con éxito
Documentación externa	Implementado con éxito

Manual de usuario

Configuración

Para poder configurar esta aplicación se debe instalar Rails y Ruby. Par eso se recomienda seguir los pasos de este tutorial: http://ruby.railstutorial.org/chapters/, en los cuales se explica fácilmente cómo lograr obtener estas herramientas.

También se debe instalar la gem Omniauth, siguiendo las instrucciones de esta dirección: https://github.com/arunagw/omniauth-twitter.

Lo anterior, es si se desea tener un mayor control de la aplicación, de lo contrario, con un dispositivo con acceso a internet configurado se puede hacer acceso a la aplicación desde http://nuevento.herokuapp.com

Simplemente se debe tener un ordenador con el acceso a Internet configurado.

Instalación

Se deben colocar los archivos encontrados en http:github.com/julioagueros/appEventos en una carpeta del filesystem para poder usarlos por medio de los comandos de Rails.

Para poder correr el servidor de Rails, se debe escribir rails s en una ventana del Terminal, a partir de ahí, abrir un browser de internet y dirigirse a la dirección localhost:3000.

Si se quiere utilizar la aplicación desde heroku, entonces simplemente se debe realizar la instalación de algún web browser como lo es Google Chrome, Mozilla Firefox o Opera. Se recomienda que sea una versión actualizada.

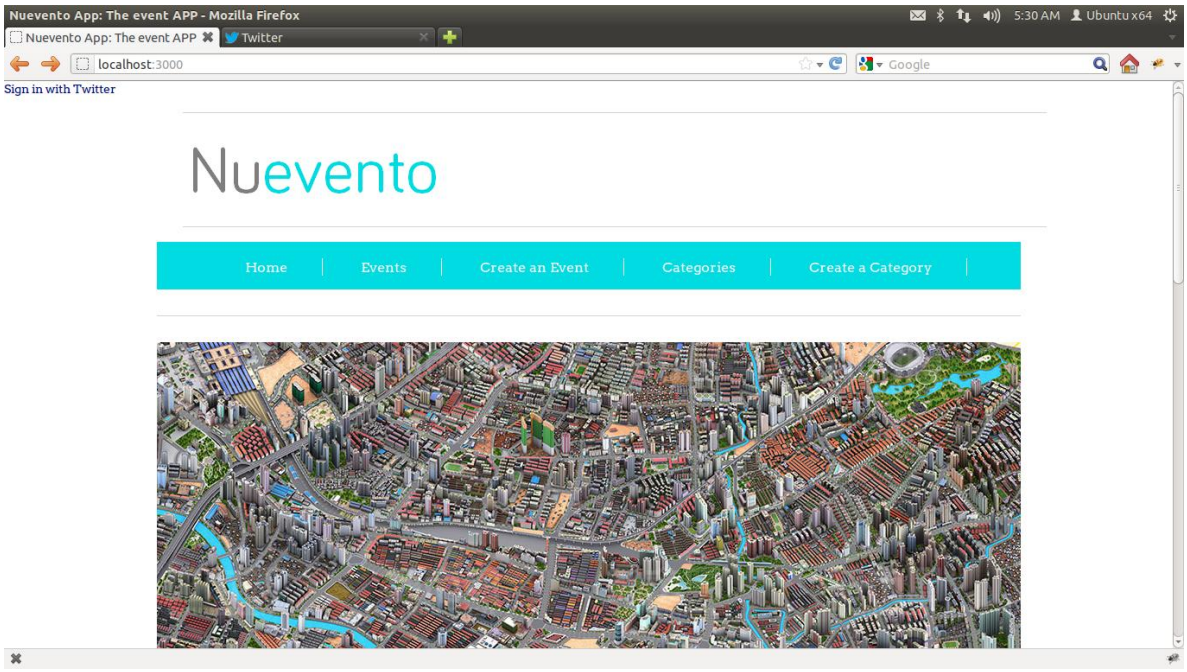
Uso

Se debe crear una cuenta en Twitter en: http://twitter.com

Opción 1. Abrir una ventana del explorador y dirigirse a la página web http://nuevento.herokuapp.com y a partir de ahí se puede navegar a través del menú que se muestra para eso.

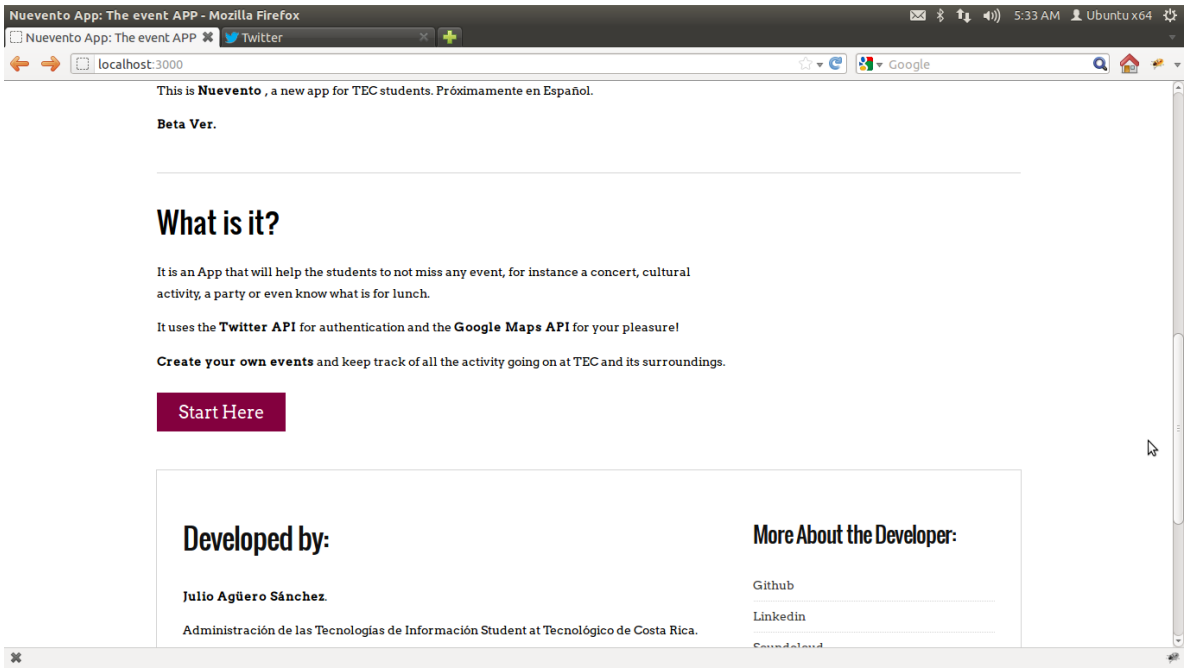
Opción 2. Realizar el procedimiento anterior para configurar e instalar la aplicación y correr el servidor de rails como se especificó anteriormente.

A partir de ahí, se verá una página como esta:

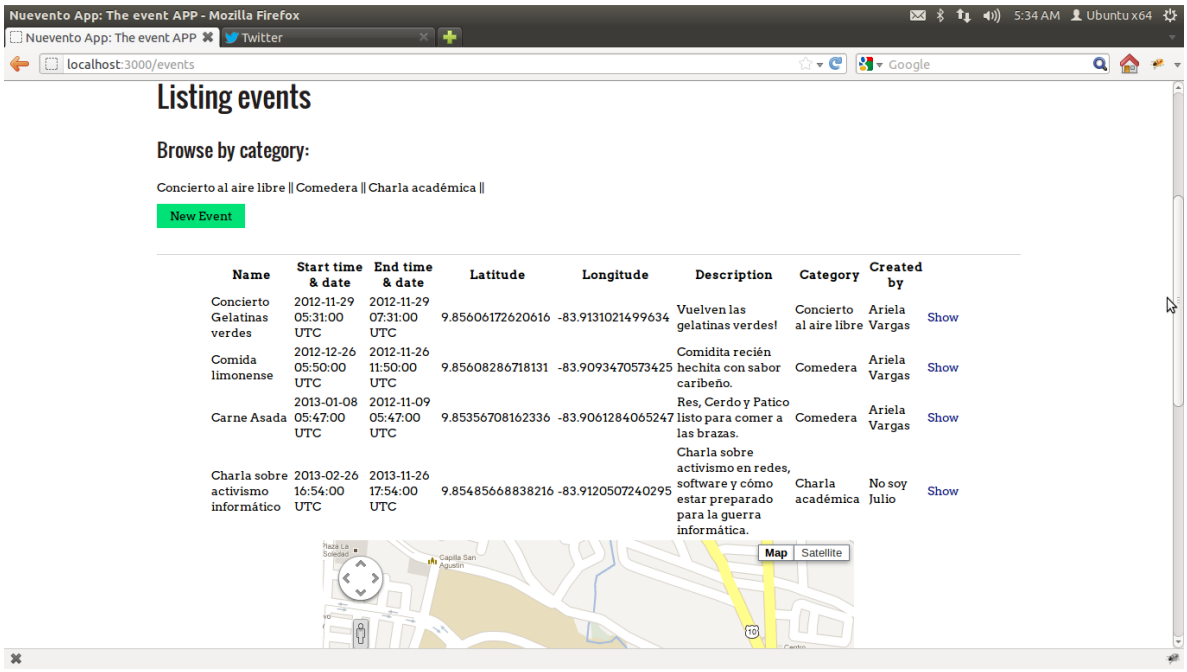


Como se nota, se tiene un menú en la parte superior, por medio de este se puede navegar a través de toda la página.

En esta misma página de bienvenida se tiene este botón de "Start Here":

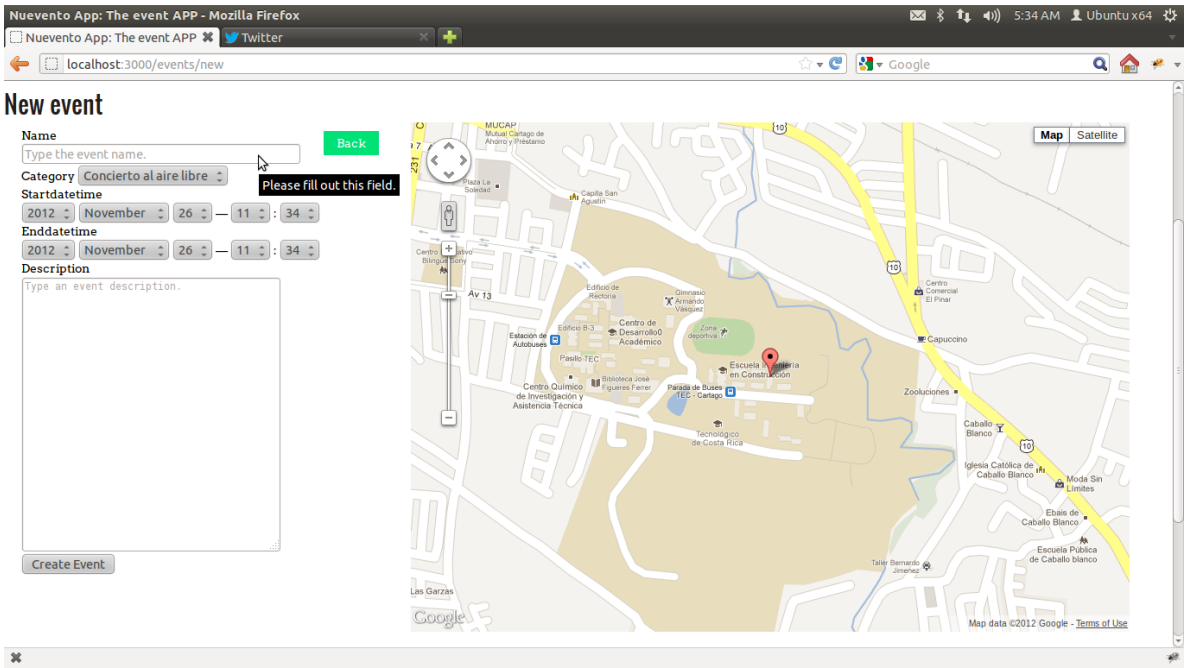


Si se hace click en este se llega al index de los eventos:



En esta vista se pueden ver todos los eventos que se tienen. El usuario solamente se puede editar o borrar los eventos que él o ella ha creado.

Si se hace click en "new event", se entrará a la funcionalidad de crear un nuevo evento:



En esta vista, se ingresan los datos del evento, se selecciona un punto en el mapa de la ubicación del evento y se le da click a "create event" y listo, se ha creado un evento.

Conclusiones

- Realizar un trabajo que uno mismo propuso es sin duda alguna, un gran motivante para realizarlo con más ganas y querer mejorarlo inclusive hasta después de entregado académicamente
- Para el API de Twitter se debe tener cuidado con lo que es la sincronización. Por ejemplo si el reloj del sistema no está sincronizado con la realidad, se da un error 401 de autorización.
- Es recomendable investigar qué maneras hay de realizar las cosas antes de realizarlas, con esto quiero decir que en Ruby existen "gems" que facilitan el trabajo y hacen que no se "reinvente la rueda" y además encierran buenas prácticas y han sido muy probadas, por lo que tienen gran confiabilidad.
- Utilizar herramientas como el scaffold ahorran mucho tiempo de desarrollo. Al crear este "esqueleto", se puede basarse en ello y desarrollar a partir de esto.
- Las aplicaciones web tienen una gran ventaja ya que no están ligadas a una plataforma específica. Por lo que se puede hacer uso de la aplicación con sólo tener un web browser.
- El uso de un framework permite tener estandarización hasta cierto punto, por las convenciones sobre nombres, la estructura de la aplicación y hasta la estructura de las carpetas en las que se contiene el código fuente.
- El uso de un sistema de control de versiones distribuido es de gran ayuda para tener un seguimiento de las acciones realizadas en la aplicación. Inclusive si se realizó un error, se puede regresar a una versión funcional sin mayor problema.
- Lograr el deployment de una aplicación es una gran ventaja para que el usuario pueda hacer uso de la misma desde cualquier computadora con acceso a Internet.
- Hay que tener claro qué versiones de tecnologías se están usando ya que los comandos y funciones se van depreciando y paran de ser usadas, por lo que se puede estar revisando una fuente desactualizada y probando funciones que ya no existen en el lenguaje o tecnología.
- Al no encontrar mucha documentación sobre errores, hay que dirigirse a fuentes no oficiales por lo que se debe ser muy crítico a la hora elegir cuál fuente es la correcta para poder basarse en ella y resolver los problemas.