Protege la identidad de quienes visitan la página y aquellos que suben información.
El sistema NO RECOPILA datos de **visitantes** (quienes ingresan al Sistema solo para consultar la información disponible).

Por instancia vamos a tener una o más ***fuentes de datos***, donde se almacenará información disponible.

FUENTES ESTÁTICAS
Proveen información que se obtiene de ***datasets***. Brinda servicio de lectura para publicar información estática, usando los datasets.
FUENTES DINÁMICAS
Permite que las personas carguen hechos, en formato de texto, imagen, audio o video.
Brinda servicio para la subida de información por perfiles ***anónimos*** y/o ***registrados***.
FUENTES PROXY
Integraciones con servicios de ***fuentes de datos*** y ***datasets*** provistos por otras ONGs.
//no lo entendi

—----------

Contará con:

- Servicio de ***agregación***, donde se va a consultar a las distintas fuentes para combinar sus datos.
- Servicio de ***visualizaciones*** *y* ***estadísticas***.
- Módulo para la **recepción de denuncias** sobre el contenido y protección de datos personales.

CLASES: atributos - métodos

- **Colección**: título (VC), descripción(VC), fuente (Fuente) y criterio (VC) - Métodos:
    - hechos(): trae hechos de la fuente, y los filtra mediante el criterio.
    - eliminarColeccion(persona): eliminación manual de la colección, esta debe analizar si aquel que intenta eliminarla es un …
    - cambiarCriterio(): cambia el criterio de pertenencia de la colección, ya que este es configurable.
- **Hecho**: etiqueta (VC), título (VC), descripción (VC), categoría (VC), lugar_acontecimiento (ubicación) y fecha_acontecimiento(DT), fecha_carga(DT), origen y tipo - Metodos:
    - agregarMultimedia(archivoMultimedia): cambia el atributo tipo a “multimedia” y le manda mensaje la clase “Multimedia” para que agregue el archivo.
    - etiquetar(nuevaEtiqueta): da valor al atributo “etiqueta”. Solo los administradores son capaces de etiquetar
- **Multimedia**: tipo de hecho el cual hereda todos los atributos, y además tiene archivosMultimedia (lista de archivosMultimedia) - Métodos:
    - agregarMultimedia(archivos Multimedia): añade el archivo multimedia llegado por parámetro a la lista.
- **Ubicación**: Representación de latitud y longitud en el mapa.

—----------

ACLARACIONES: Franco

- **Colección**
  - "Las colecciones ***están asociadas a una fuente y tomarán los hechos*** de las mismas: para esto las colecciones también contarán con un ***criterio de pertenencia*** configurable"
    - El Usuario va a seleccionar *"Incendios Forestales"*, que va a ser la categoría/criterio de pertenencia de la colección. Con este criterio, se hace un filter en la fuente donde están almacenados los hechos mediante el método getHechos(Categoría categoría);.
- **Hecho**
  - La categoría del Hecho va a ser un ***enum*** para que se pueda filtrar por categoría en la base de datos de todos los hechos.
  - "Hasta el momento, ***existen dos tipos de hechos***: de texto y con soporte para contenido multimedia. Para esta primera iteración, sólo se requerirá dar soporte al primero, ***pero es importante contemplarlo en el diseño de la plataforma***."
    - Hacer la clase "*Hecho*" abstracta para que luego los diferentes tipos de Hechos la usen con Herencia.
- **<<Interface>> Criterio -** STRATEGY
  - "... las colecciones también contarán con un criterio de pertenencia ***configurable***"
  - La interfaz ***Criterio*** sería la ***estrategia*** y las diferentes formas de filtrar en una base de datos van a ser las ***estrategias concretas***. El Usuario va a ser el que diga que estrategia en concreto va a utilizar, por ejemplo: filtrar por Categoría, Lugar, por Año y Lugar, por Año, Categoría y Lugar, etc.
- **<<Abstract>> Fuente -** TEMPLATE
  - Siempre se va a tener que abir el archivo, almacenar sus hechos, y cerrar el archivo. Eso sería la plantilla.
  - Cada forma de obtener los hechos de esas fuentes van a ser las estrategias en concreto. En este caso se pide dar soporte para archivos CSV. FuenteCSV va a definir su manera de leer y guardar los hechos según el formato.

Diagramas

- Diagrama de Clase
- Diagrama de Despliegue
- Diagrama de Casos de Uso

Decisiones de diseño:

- Pusimos una clase Hecho ya que todavía no hay información de Hecho Multimedia.
- En vez de hacer una clase Ubicación, pusimos latitud y longitud como atributos. Ubicación no tendría comportamiento por eso no es una clase.

- Pusimos Categoría como enum para dejar categorías predeterminadas y poder filtrar en la Colección de Hechos.
- Decidimos utilizar el patrón "***Strategy***" con criterio para poder encapsular cada tipo de criterio particular y permitir cambiar el criterio de pertenencia, ya que es configurable.
- Decidimos utilizar el patrón "***Template***" con fuente ya que siempre se va a tener que abrir el archivo, almacenar sus hechos, y cerrar el archivo. Esa sería la plantilla. Cada forma de obtener los hechos de esas fuentes van a ser las estrategias en concreto.
- Validamos que cada atributo del constructor de hecho y colección no se cargue vacío para que no hayan inconsistencias.

Duda:

1) El constructor de hecho seria mejor hacerlo con un builder?