UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

CÁTEDRA
DISEÑO DE SISTEMAS
DE INFORMACIÓN

MetaMapa: Sistema de gestión de mapeos colaborativos

Trabajo Práctico Anual Integrador
-2025-

Contexto general

Contexto

La inteligencia colectiva permite solucionar problemas a partir del aporte de cada individuo de información de la que dispone. Sin embargo, para que esta sea confiable, es importante poder ordenarla, etiquetarla y ubicarla en tiempo y espacio. Esto permitirá, por ejemplo, en una ubicación determinada, reportar focos de incendios y prevenirlos, visibilizar causas, identificar focos de contaminación o aportar información valiosa para resolver desapariciones forzadas. Para estos fines, contar con información multimedia adecuadamente organizada en tiempo y geográficamente puede ser fundamental.

Nuestro sistema

MetaMapa será un Sistema de código abierto encargado por una ONG para la recopilación, visibilización y mapeo colaborativo de ***información***. *MetaMapa* estará diseñado para que otras ONG, universidades u organismos estatales puedan instalarlo en sus servidores, gestionarlo y ofrecerlo para sus comunidades, en forma de instancias particulares. El sistema debe rendir cuentas[1] a la sociedad toda, maximizando la disponibilidad y veracidad de la información, protegiendo la identidad de quienes lo visitan y suben información. Por eso, *MetaMapa,* más allá de estadísticas básicas de uso, no recopilará datos de visitantes, es decir, quienes ingresan al Sistema únicamente para consultar la información disponible.

Con el objetivo de garantizar la disponibilidad de información y mitigar fallas de infraestructura y posibles ataques deliberados, esta se almacenará de forma descentralizada: cada instancia de *MetaMapa* podrá contar con una o más ***fuentes de datos,*** servidas desde diferentes nodos***,*** en las que se almacenará la información disponible en sí. Esta información podrá ser redundante, y es uno de los desafíos de *MetaMapa* proveer herramientas de ***etiquetado*** que permitan vincular *la* ***información*** y generar consenso sobre la misma.

Algunas de estas ***fuentes de datos*** serán **estáticas**, es decir, proveerán información obtenida directamente de ***lotes de datos*** (*datasets*) bien conocidos. Sin embargo, *MetaMapa* también permitirá que cualquier persona mayor de edad pueda subir piezas de información (llamadas ***hechos***), ya sean descripciones textuales, imágenes, audios o videos. Para esto, también se desarrollarán fuentes de datos **dinámicas**, que permitirán que las personas carguen colaborativamente nuevas piezas de información.

En este marco, diseñaremos e implementaremos:

- *una* ***fuente estática***: un servicio de solo lectura para publicar información estática, utilizando datasets que provengan de archivos u otras fuentes de datos provistas por otras ONGs,
- *una* ***fuente dinámica***: un servicio propio para la subida colaborativa de información, por perfiles anónimos y/o registrados,
- *una* ***fuente proxy***: integraciones con servicios de ***fuentes de datos*** y ***datasets*** provistos por otras ONGs.
- un servicio de ***agregación***, que se encargará de consultar las distintas ***fuentes de datos*** *y* combinar sus datos,
- un servicio de ***visualizaciones*** *y* ***estadísticas***, que podrán ser consultadas tanto online a través de una interfaz Web como consumidas por otras ONGs,
- un módulo para la **recepción de denuncias** sobre el contenido y protección de datos personales.

[1] https://www.researchgate.net/publication/221248162_Designing_for_accountability

A modo orientativo, se tiene este diagrama de despliegue (Fig. 1) como contexto general del sistema. El mismo no es definitivo y sufrirá modificaciones conforme avance el desarrollo del sistema. Además, no todas las instituciones tendrán por qué seguir esta misma estructura de despliegue, ya que la podrán ajustar a sus necesidades y a las de sus comunidades. Por ejemplo, algunas instituciones podrían exponer múltiples fuentes estáticas o consumir múltiples datasets, o sólo exponer una fuente dinámica.

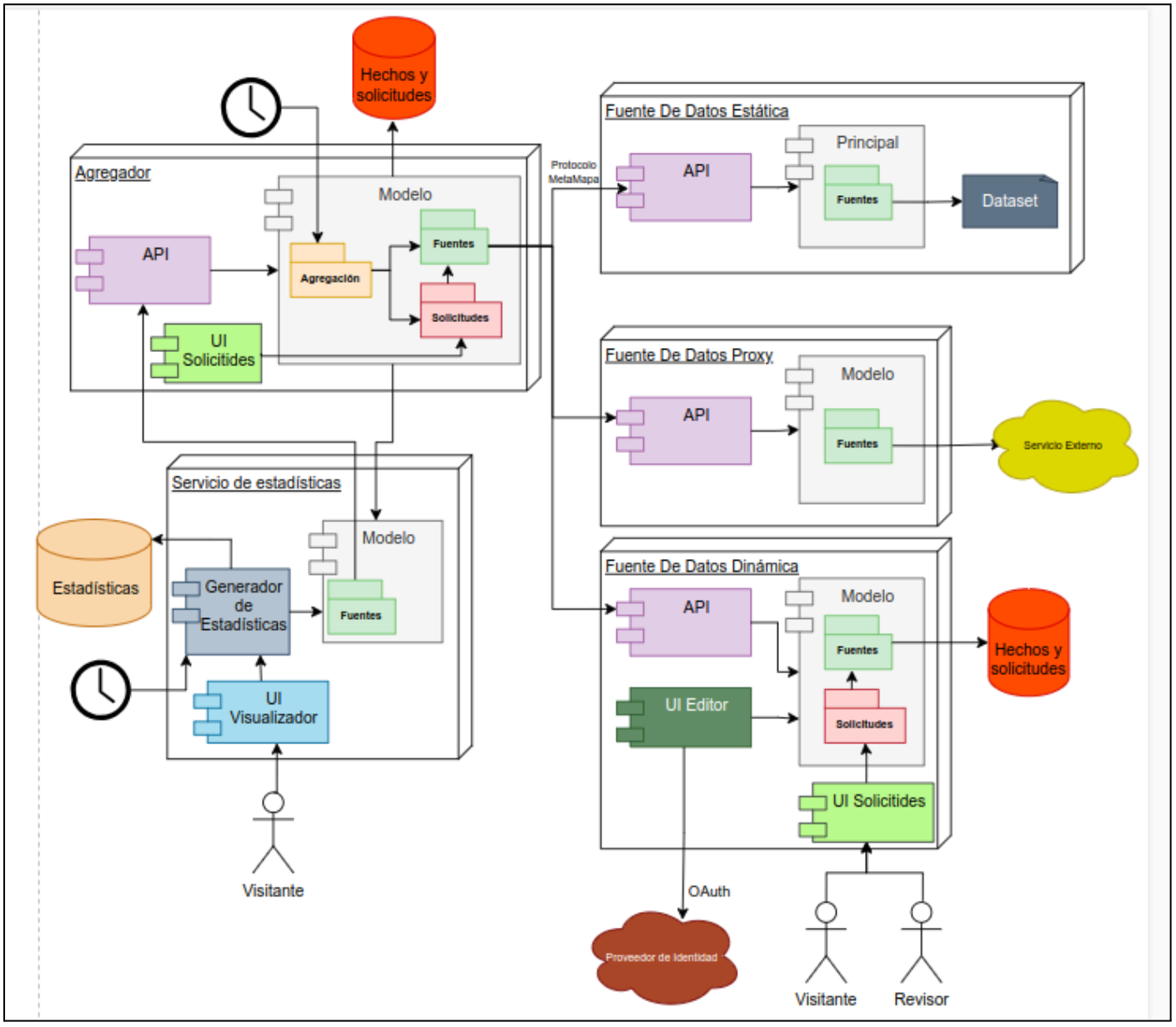


Figura 1 - Diagrama de Despliegue Inicial.

Entregas

El trabajo se encuentra estructurado en 6 entregas, algunas orientadas específicamente a la inclusión de funcionalidades, mientras que otras se abocarán a la inclusión de algunos aspectos del diseño y herramientas tecnológicas para la implementación del mismo.

Las entregas previstas se muestran a continuación, aunque pueden sufrir algunas modificaciones en su alcance o fechas:

Nro.	Título	Semana de entrega propuesta

Glosario

- **Colecciones**: conjuntos de hechos organizados bajo un título y descripción, creados y gestionados por administradores. Son públicas y no pueden ser editadas ni eliminadas manualmente.
- **Contribuyentes**: personas que suben hechos a la plataforma, ya sean anónimas o registradas con datos personales.
- **Dataset**: lote de datos bien conocidos y usualmente en formato de archivo CSV desde el cual la plataforma extrae hechos.
- **Datos dinámicos**: información proveniente de contribuciones colaborativas, como descripciones textuales, imágenes, audios o videos, que crece en cantidad a lo largo del tiempo.
- **Datos estáticos**: información proveniente de archivos o bases de datos conocidas, que usualmente no cambian a lo largo del tiempo.
- **Etiqueta**: metadato asignado a un hecho para facilitar su organización y filtrado.
- **Fuentes de datos:** orígenes de la información en el sistema, pudiendo ser estáticas (datasets), dinámicas (aportadas por usuarios) o intermediarias (**proxy,** integraciones con otras instancias de MetaMapa u otros sistemas)
- **Hecho**: pieza de información registrada en el sistema, con un título, descripción, categoría, ubicación, fecha y origen.
- **Instancia**: un caso particular de despliegue de los servicios de MetaMapa en los servidores de una ONG, estado o universidad. Este despliegue podría realizarse a través de uno o más nodos, siguiendo la arquitectura orientativa de la Figura 1, o bien podría seguir otra estructura, según las necesidades de la institución.
- **Nodos**: cada una de las computadoras que procesan, exponen y almacenan información en el sistema.
- **Servicios**: cada uno de los procesos software que procesan, exponen y almacenan información en el sistema y se ejecutan en un nodo.
- **Pieza de información**: cada una de las unidades discretas de información que una fuente expone (y que en el caso de las fuentes dinámicas se pueden cargar colaborativamente). En MetaMapa, todas las piezas de información representan hechos ubicados en un tiempo y espacio determinado.
- **Servicio de agregación**: servicio encargado de combinar datos de distintas fuentes para generar una visión integrada de la información. Los servicios de agregación exponen la información siguiendo los mismos protocolos y formatos que las demás fuentes, por lo que también pueden ser utilizados como fuentes.
- **Sistema descentralizado**: arquitectura que almacena información en distintos nodos para mejorar la disponibilidad y seguridad de los datos.

ENTREGA 1: Arquitectura y Modelado en Objetos - Parte I

Objetivos de la entrega

- Entrar en contacto con el dominio y sus principales abstracciones.
- Incorporar de forma paulatina conceptos y principios de Diseño.
- Familiarizarse con el entorno de desarrollo y las principales tecnologías a ser aplicadas a lo largo del Trabajo Práctico.
- Familiarizarse con la arquitectura del sistema.

Unidades del Programa Vinculadas

- Unidad 2: Herramientas de Concepción y Comunicación del Diseño
- Unidad 3: Diseño con Objetos
- Unidad 6: Diseño de Arquitectura
- Unidad 8: Validación del Diseño

Alcance

- Fuentes estáticas.
- Colecciones.
- Hechos: listado, filtrado y búsqueda.
- Visualizadores y contribuyentes de hechos.
- Solicitudes de eliminación.

Dominio

Fuentes estáticas

Tal como se mencionó en el detalle de nuestro sistema, se tienen ***fuentes*** de datos de tipo estáticas, provenientes de lotes de datos bien conocidos. Para esta iteración, se requiere diseñar e implementar el componente que posibilite la lectura de estos ***datasets*** y que extraiga los ***hechos*** de los mismos. En esta primera iteración estaremos incorporando un lote de datos estático de tipo archivo *.csv.*

Colecciones

Las ***colecciones*** representan conjuntos de ***hechos***. Las mismas pueden ser consultadas por cualquier persona, de forma pública, y no pueden ser editadas ni eliminadas manualmente (esto último, con una sola excepción, ver más adelante).

Las ***colecciones*** tienen un título, como por ejemplo “Desapariciones vinculadas a crímenes de odio”, o “Incendios forestales en Argentina 2025” y una descripción. Las personas administradoras pueden crear tantas ***colecciones*** como deseen.

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, que dictará si un hecho pertenece o no a las mismas. Por ejemplo, la colección de “Incendios forestales…” deberá incluir

automáticamente todos los ***hechos*** de ***categoría*** "Incendio forestal" ocurrido en Argentina, acontecido entre el 1 de enero de 2025 a las 0:00 y el 31 de diciembre de 20205 a las 23:59.

Hechos

Cada ***hecho*** representa una pieza de información, la cual debe contener mínimamente: título, descripción, categoría, contenido multimedia opcional, lugar y fecha del acontecimiento, fecha de carga y su origen (carga manual, proveniente de un dataset o provisto por un ***contribuyente***). Idealmente, todos estos campos podrán ser utilizados por cualquier ***visitante*** como filtros de búsqueda desde la interfaz gráfica y como ***criterios de pertenencia*** a una colección.

Hasta el momento, existen dos tipos de hechos: de texto y con soporte para contenido multimedia. Para esta primera iteración, solo se requerirá dar soporte al primero, pero es importante contemplarlo en el diseño de la plataforma.

Cualquier ***hecho*** provisto por ***contribuyentes*** de la plataforma podrá ser sometido a etapas de etiquetado y revisión manual por parte de las personas administradoras (lo cual será desarrollado en entregas venideras)

Obtención de hechos en lote por archivos CSV

Si bien una fuente de datos estática podría venir de cualquier tipo de dataset, se pide dar soporte al acceso de hechos mediante archivos *.csv*. Las piezas de información provenientes de este tipo de archivos se leerán directamente y no se traerán a memoria. En esta etapa, se considerará que un hecho está repetido si el "título" es el mismo, de ser así, se pisaran los atributos del existente. La ubicación del hecho vendrá determinada por su latitud y longitud, para poder representarla en un mapa en entregas posteriores.

Se deberá leer este tipo de archivos teniendo en cuenta este formato:

Título	Descripción	Categoría	Latitud	Longitud	Fecha del hecho

Visualizadores y contribuyentes de hechos

Se considera **visualizador** a toda persona humana anónima que desee ingresar a la plataforma para visualizar y utilizar la información disponible en el Sistema. No requerirá identificarse, y podrá subir hechos si así lo quisiera manteniendo su anonimato. Esto se debe a que al mostrar en el Sistema los hechos aportados por este tipo de personas usuarias, no se mostrará dato alguno de su identidad. La información de quién subió un determinado hecho quedará disponible en caso de que sea necesario una revisión sobre ellos.

En cambio, si lo desearan podrían darse a conocer y subir hechos a los que se les asocie algunos datos personales, como nombre, apellido y edad. Únicamente el nombre sería el campo obligatorio.

Consideraremos a las personas humanas que han aportado alguna pieza de información como ***contribuyentes***.

Solicitudes de eliminación

Cualquier ***hecho*** del sistema debe admitir solicitudes de eliminación, para aquellas situaciones adecuadamente fundadas (por ahora, mediante un texto de al menos 500 caracteres) en que se deba eliminar del sitio la información, aún cuando esté en una fuente.

Cuando estas ***solicitudes*** sean aceptadas, el hecho se mantendrá internamente en el sistema, pero ya no se mostrará en ninguna colección y no se lo incorporará nuevamente si una fuente lo vuelve a retornar.

Naturalmente muchas ***solicitudes*** serán *spam*, por lo que las mismas quedarán pendientes hasta que una persona administradora las acepte o rechace (eliminándose por completo).

Requerimientos

Requerimientos de dominio

1. Como persona ***administradora***, deseo crear una colección.
2. Como persona ***administradora***, deseo poder importar hechos desde un archivo CSV. [2]
3. Como persona ***visualizadora***, deseo navegar todos los hechos disponibles de una colección.
4. Como persona ***visualizadora***, deseo navegar los hechos disponibles de una colección, aplicando filtros.
5. Como persona ***contribuyente***, deseo poder solicitar la eliminación de un hecho.
6. Como persona ***administradora***, deseo poder aceptar o rechazar la solicitud de eliminación de un hecho.

Entregables

1. **Modelo del Dominio:** diagrama de clases inicial que contemple las funcionalidades requeridas.
2. **Justificaciones de Diseño Iniciales.**
3. **Diagrama General de Casos de Uso.**
4. **Diagramas de arquitectura**: diagrama de despliegue, componentes y/o cualquier otro tipo de diagrama que refleje la arquitectura física y lógica de la entrega actual.
5. **Implementación** de los requerimientos.

[2] Se debe utilizar una fuente que considere más de 10.000 hechos en cada importación. Las fuentes propuestas son de ejemplo, se puede proponer una propia.

ENTREGA 2: Arquitectura, y Modelado en Objetos - Parte II

Objetivos de la entrega

- Diseñar e implementar, de manera incremental, las nuevas funcionalidades.
- Incorporar nociones de ejecuciones de tareas asincrónicas y/o calendarizadas.
- Familiarizarse con otros componentes dentro de la arquitectura del Sistema y la orientación a servicios.
- Familiarizarse con diferentes estilos de integración con componentes y servicios externos
- Familiarizarse con las APIs y su consumo.

Unidades del Programa Vinculadas

- Unidad 2: Herramientas de Concepción y Comunicación del Diseño
- Unidad 3: Diseño con Objetos
- Unidad 6: Diseño de Arquitectura
- Unidad 8: Validación del Diseño

Alcance

- Fuentes dinámicas.
- Fuentes proxy (intermediarias). Integración con sistemas externos
    - de otras ONGs
    - de las propias instancias de MetaMapa.
- Rechazo de solicitudes de eliminación por tratarse de *spam* en forma automática.

Dominio

Fuentes dinámicas

En esta iteración, se implementará la funcionalidad que permitirá a los ***contribuyentes*** del Sistema subir ***hechos*** en forma anónima o registrada. Retomando el diagrama de arquitectura provisto en el Contexto general, este requerimiento deberá satisfacerse en el servicio de ***fuentes dinámicas***[3]. Si la persona no está registrada en la plataforma, podrá subir hechos sin posibilidad de edición posterior. En cambio, si los sube en forma registrada podrá realizar modificaciones al mismo en caso de que lo necesitara, pero solo en el plazo de una semana.

En ambos casos podrán estar sujetos a revisión por parte de las personas ***administradoras,*** que podrán aceptar, aceptar con sugerencia de cambios o rechazar la información. Se desarrollarán los mecanismos necesarios para gestionar la subida de información en formato tanto de texto como multimedia.

Fuentes proxy (intermediarias)

Existen sistemas externos provistos por otras organizaciones no gubernamentales que proveen hechos de generación propia mediante APIs y/o bibliotecas de diferente tipo. Aunque cada una de estas ***fuentes externas*** presentarán características propias y formatos diferentes, nos interesa diseñar un mecanismo

[3] En el diagrama de despliegue, se indica que el servicio de fuentes dinámicas tiene una base de datos de Hechos y Solicitudes. Como aún no tratamos bases de datos en la cursada, recomendamos implementar un Patrón Repositorio que cumpla esa función y mantenga las entidades necesarias en memoria.

general para integrarnos con ellas de forma unificada. Tal como se observa en el diagrama de arquitectura, esto se realizará a través de ***servicios*** y ***fuentes proxy*** (intermediarias).

Para poner a prueba este enfoque, deberemos dar soporte a dos nuevos componentes:

- **Fuente Demo:** una fuente que pueda dialogar con un sistema externo prototípico (y ficticio). En otras palabras, se trata de una integración con un sistema externo ficticio, para el cual contamos con una biblioteca cliente.
- **Fuente MetaMapa:** una fuente que nos permita dialogar con otras instancias de MetaMapa, mediante una interfaz REST.

Fuente Demo:

Dado que estas fuentes externas pueden tener interfaces muy diferentes entre sí —por ejemplo, bibliotecas complejas o APIs REST—, el objetivo es diseñar un componente flexible y extensible que pueda adaptarse a estos distintos modos de acceso y facilitar la obtención de hechos desde servicios externos.

Para cumplir con este objetivo, daremos soporte a un componente nuevo denominado ***fuente demo***, que parte de la siguiente interfaz:

```
 /** Hasta donde sabemos, no existe ninguna fuente externa que tenga esta interfaz. La misma es sólo a  que  modo de ejemplo y para poner nuestro diseño a prueba */
class Conexion {
   /**
    * Devuelve un mapa con los atributos de un hecho, indexados por nombre de
    * atributo. Si el método retorna null, significa que no hay nuevos hechos
    * por ahora. La fecha es opcional
   */
   Map<String, Object> siguienteHecho(URL url, DateTime fechaUltimaConsulta)
}
```

Es posible (y deseable, como se verá en futuras entregas) que más de una fuente retorne el mismo hecho, es decir, con iguales o muy similares atributos.

Debido a que este servicio externo es stateful, es decir, sus resultados dependen de los resultados previos, necesitamos que, una vez por hora, esta fuente proxy revise e incorpore nuevos ***hechos***, si hubiera. En otras palabras, el servicio externo no se consumirá en tiempo real.

Fuente MetaMapa:

Por otro lado, el Sistema deberá proveer la capacidad de comunicarse con otras instancias de MetaMapa. Esto lo haremos a través de una interfaz API REST, siguiendo la siguiente estructura tentativa (con posibles modificaciones en el futuro):

Endpoint	**Descripción**
GET /hechos	Esta ruta expone todos los hechos del sistema y los devuelve como una lista en formato JSON. La misma acepta parámetros para filtrar los

	resultados: categoria, fecha_reporte_desde, fecha_reporte_hasta, fecha_acontecimiento_desde, fecha_acontecimiento_hasta, ubicacion.
GET /colecciones/:identificador/hechos	Esta ruta, similar a la anterior, permite obtener los hechos asociados a una colección. Acepta los mismos parámetros y devuelve los resultados en el mismo formato.
POST /solicitudes	Permite crear solicitudes de eliminación, enviando los datos de la solicitud como un JSON a través del cuerpo *(body)* de la misma

Si bien en próximas iteraciones trabajaremos en exponer nuestros propios servicios utilizando esta interfaz, **por ahora se desea que nuestras fuentes proxy también sean capaces de consumir este tipo de API.** A diferencia del servicio anterior, este será consumido en tiempo real y se espera que sus resultados sean siempre actuales.

Colecciones

Para esta iteración, sigue válido el desarrollo de las colecciones realizado en la entrega anterior, pero se debe contemplar que a partir de esta entrega las colecciones podrán surgir también por fuentes dinámicas o fuentes proxy.

Además, para posibilitar la futura exposición REST de las mismas, se incorporará un atributo textual identificador ***(handle)*** que será un string alfanumérico, sin espacios, único entre todas las colecciones de un mismo servicio.

Rechazo de solicitudes de eliminación spam en forma automática

En la anterior entrega, se le permitió a las personas administradoras aceptar o rechazar las solicitudes de eliminación creadas por visualizadores y contribuyentes en forma manual. Para simplificar la tarea, se cuenta con un componente ya desarrollado que permite la detección de casos de spam evidentes. Si éste marca a la solicitud como *spam*, la solicitud deberá ser rechazada automáticamente.

```
interface DetectorDeSpam {
   boolean esSpam(String texto)
}
```

Requerimientos detallados

1. Como ***persona contribuyente***, deseo poder crear un hecho a partir de una fuente dinámica.
2. Como ***persona usuaria,*** quiero poder obtener todos los hechos de una fuente proxy demo configurada en una colección, con un nivel de antigüedad máximo de una hora.
3. Como persona usuaria, quiero poder obtener todos los hechos de las fuentes MetaMapa configuradas en cada colección, en tiempo real.

4. El Sistema debe permitir el rechazo de solicitudes de eliminación en forma automática cuando se detecta que se trata de spam.

Entregables

1. **Modelo del Dominio:** diagrama de clases que contemple las funcionalidades requeridas
2. **Justificaciones de Diseño**: Documento y Diagramas Complementarios.
3. **Implementación** de requerimientos de la Entrega Actual. Favorecer la reutilización del código de lo ya desarrollado.
4. **Diagrama de arquitectura** actualizado.
5. **Respuestas** a las siguientes preguntas, a modo de debate:
    - ¿Cómo se podría implementar un filtro de spam sin recurrir a servicios externos? Sugerencia: investigar sobre *el algoritmo TF-IDF para la detección de spam*.
    - ¿Cómo se podría implementar un filtro de spam en base a consumir servicios externos en la nube? ¿Qué consecuencias desde el punto de vista de costos y seguridad de datos traería?
6. **Bonus**: implementación de un filtro de spam básico que no requiera de un sistema externo.

ENTREGA 3: Integración, y Modelado en Objetos - Parte III

Objetivos de la entrega

- Diseñar e implementar, de manera incremental, las nuevas funcionalidades.
- Exponer un servicio a través de un protocolo de red.
- Incorporar flujos de trabajo asincrónicos.

Alcance

- Implementación de una fuente propia REST
- Integración con una fuente propia
- Servicio agregador
- Agregación de fuentes y algoritmos de consenso.

Unidades del Programa Vinculadas

- Unidad 2: Herramientas de Concepción y Comunicación del Diseño
- Unidad 3: Diseño con Objetos
- Unidad 6: Diseño de Arquitectura
- Unidad 8: Validación del Diseño

Dominio

Servicio de agregación

Para esta iteración, se requiere modelar el servicio de agregación descripto en el diagrama de despliegue inicial. Esto permitirá que las personas visitantes y/o contribuyentes de la plataforma accedan a hechos provenientes de todas las fuentes. Por lo tanto, se permite ahora que las colecciones contengan hechos de distintas fuentes, sean estas estáticas, dinámicas o provenientes de servicios externos (fuentes proxy).

Se pide que, una vez por hora, el servicio de agregación lea las fuentes disponibles, para incorporar nuevos hechos, si hubiera. Se deberán contemplar las integraciones necesarias en base al diagrama de arquitectura. Esto implica leer las fuentes y replicar las piezas de información que se hubieran incorporado en el período desde la última integración (o desde el inicio, en caso de que se tratara de la primera vez).

Modos de navegación

En iteraciones anteriores, cualquier persona podía navegar las colecciones, aplicando opcionalmente filtros. Sin embargo, en esta entrega se incorpora el concepto de ***modo de navegación***, que le permitirá elegir qué hechos se mostrarán**:**

- de forma ***irrestricta***, en la cual navegarán todos los hechos, sin ningún tipo de curación;
- de forma ***curada***, en la cual se navegarán sólo los hechos para los que se tiene ***consenso***.

Consenso

Para reducir el ruido que podría darse por presentar hechos dudosos o falsos, al crear una colección, se podrá especificar opcionalmente un algoritmo de ***consenso***. Estos son algunos de los que se implementarán:

- múltiples menciones: si al menos dos fuentes contienen un mismo hecho y ninguna otra fuente contiene otro de igual título pero diferentes atributos, se lo considera consensuado;
- mayoría simple: si al menos la mitad de las fuentes contienen el mismo hecho, se lo considera consensuado;
- absoluta: si todas las fuentes contienen el mismo, se lo considera consensuado.

Si no se especifica un algoritmo de consenso, todos los hechos se consideran consensuados y estarán disponibles en la navegación curada. Por último, se considera que la ejecución de estos algoritmos puede ser costosa, por lo que es importante que no se ejecuten cada vez que ingresa un hecho, sino en horarios de baja carga en el sistema.

Exposición REST

Se deberán exponer las siguientes operaciones en una *API REST*:
API administrativa de MetaMapa:

- Operaciones CRUD sobre las colecciones.
- Modificación del algoritmo de consenso.
- Agregar o quitar fuentes de hechos de una colección.
- Aprobar o denegar una solicitud de eliminación de un hecho.

API pública para otras instancias de MetaMapa (ver entrega anterior):

- Consulta de hechos dentro de una colección.
- Generar una solicitud de eliminación a un hecho.
- Navegación filtrada sobre una colección.
- Navegación curada o irrestricta sobre una colección.
- Reportar un hecho.

Requerimientos de dominio:

1. Como visualizador o contribuyente, deseo poder seleccionar el modo de navegación de los hechos.
2. Como persona administradora, quiero asociar un algoritmo de consenso a una colección.
3. Como persona administradora, quiero modificar las fuentes de una colección.
4. El Sistema debe permitir la realización de operaciones a través de los *endpoints* solicitados, mediante el desarrollo de una API REST.

Entregables

1. **Modelo del Dominio**: diagrama de clases que contemple las funcionalidades requeridas.
2. **Justificaciones de Diseño**: Documento y Diagramas Complementarios.
3. **Implementación** de los requerimientos de esta entrega.
4. **Diagrama de arquitectura** actualizado.

ENTREGA 4: Persistencia

Objetivos de la entrega

- Incorporar nociones de persistencia de datos en un medio relacional.
- Incorporar nociones de la técnica de mapeo objeto – relacional.
- Incorporar nociones de desnormalizaciones del modelo relacional
- Exponer y documentar un servicio para ser utilizado por un ente externo.

Alcance

- Normalización de la información
- Persistencia del modelo de objetos previamente generado
- Soporte para incorporación de videos e imágenes
- Exportación de datos en formato CSV
- Estadísticas geográficas sobre todos los hechos disponibles. Queda a criterio del equipo como y cuando se computarán estás estadísticas. Se necesita al menos saber:
- Hechos agrupados por país, provincia, municipio y barrio.bonus: agrupación custom
- Soporte para búsqueda por texto libre

Dominio

Servicio de Estadísticas

Se desea desarrollar un servicio que, conectándose con el servicio agregador, genere periódicamente estadísticas sobre el uso del sistema y las muestre mediante una interfaz gráfica al usuario. Cada cierto intervalo de tiempo, se recalcularán las estadísticas y se actualizarán los *dashboards* de la interfaz.
Específicamente, se piden obtener datos que permitan responder las siguientes preguntas:

- De una colección, ¿en qué provincia se agrupan la mayor cantidad de hechos reportados?
- ¿Cuál es la categoría con mayor cantidad de hechos reportados?
- ¿En qué provincia se presenta la mayor cantidad de hechos de una cierta categoría?
- ¿A qué hora del día ocurren la mayor cantidad de hechos de una cierta categoría?
- ¿Cuántas solicitudes de eliminación son spam?

Piezas de Información

1. Como persona usuaria, deseo poder agregar una “pieza de información” a un hecho del tipo video e imagen.

Normalización de Hechos e Información

Requerimientos detallados

1. Se deberán persistir las entidades del modelo planteado. Para ello se debe utilizar un ORM.

2. Se deberá implementar el servicio que tuviera asignado el grupo haciendo uso de la/s tecnología/s que el docente considere.

Entregables

1. **Modelo del Dominio**: actualización del modelo de Diagrama de Clases con las funcionalidades previstas en esta entrega.
2. **Justificaciones de Diseño**: Documento y Diagramas Complementarios.
3. **Modelo de datos**: diagrama de entidad-relación físico.
4. **Justificaciones y Consideraciones de Diseño Relacional.**
5. **Implementación** en código de los requerimientos de la presente entrega.
6. **Documentación de API** de descarga de reportes.

ENTREGA 5: Arquitectura Web MVC y Maquetado de Interfaz de Usuario

Objetivos de la entrega

- Incorporar nociones de Diseño UI/UX.
- Incorporar nociones de maquetado Web a través de un lenguaje de marcado como HTML5.
- Incorporar nociones de aplicación de estilos sobre maquetas Web a través de CSS.
- Implementar de un Cliente Liviano (server side rendering)

Alcance

- Diseño y maquetado de interfaces de usuario
- Implementación de un Cliente Liviano
- Integración con un Servicio Externo

Requerimientos detallados

1. Como persona usuaria, deseo poder visualizar los hechos en un mapa
2. Como persona administradora, deseo poder iniciar sesión en el sistema
3. Como persona administradora, deseo poder configurar desde mi panel de control las fuentes y criterios de consenso
4. Como persona administradora, deseo poder aprobar o rechazar las solicitudes de eliminación.

Entregables

1. ***Wireframes*** *de las interfaces de usuario requeridas.*
2. ***Maquetado*** *de las interfaces de usuario requeridas.*
3. **Implementación** de cliente liviano
4. ***Implementación*** *en HTML de las interfaces de usuario, considerando la navegabilidad entres las interfaces.*

ENTREGA 6: Despliegue, Observabilidad y Seguridad

Objetivos de la entrega

- Familiarizarse con técnicas y proveedores de despliegue
- Incorporar herramientas de monitoreo y seguridad

Alcance

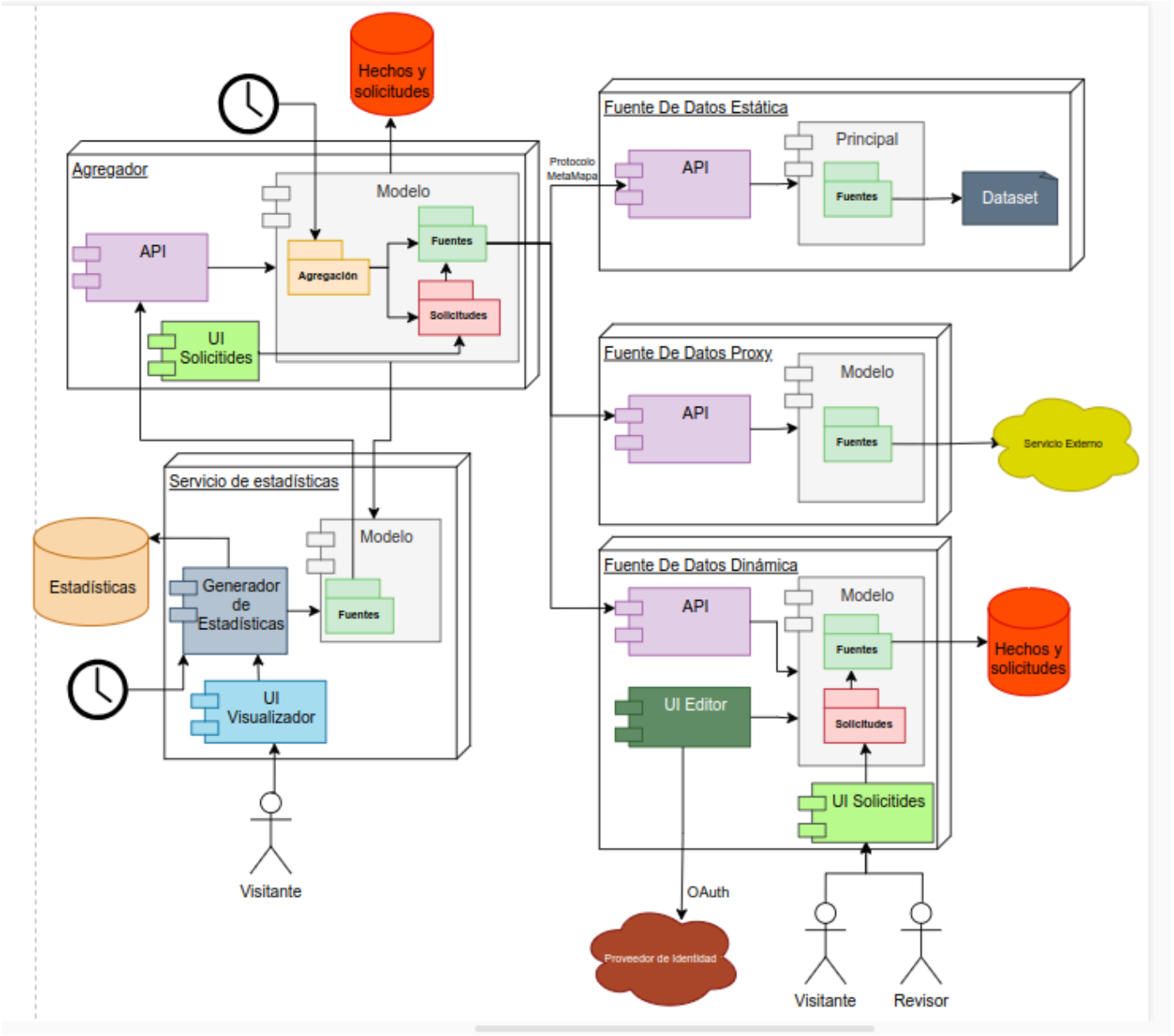
- Sistema desplegado en la nube
- Incorporar herramientas de observabilidad
- Incorporar herramientas de monitoreo

Requerimientos detallados

1. Se deberá desplegar el sistema en la nube para que pueda ser accedido por el público general.
2. Incorporar herramientas que permitan la observabilidad del sistema, de forma que sea posible conocer, a alto nivel, el estado general del sistema.
3. Incorporar herramientas que permitan el monitoreo y supervisión del sistema, de forma que el mismo se reinicie automáticamente ante un caída
4. Incorporar un sistema de Rate Limiting, que proteja el sistema
5. Incorporar un sistema de Bloqueo de IPs, que permita configurar a las que no se responderán pedidos HTTP
6. Implementación de protocolo grpc en lugar de http para tener actualizaciones en tiempo real. Nos dimos cuenta de que no escala (???

Entregables

1. **Diagrama de Despliegue** de la solución actual.
2. **Diagrama de Despliegue** de la solución propuesta.
3. **Implementación** de **Herramienta/s de Observabilidad.**


Hechos y solicitudes
Agregador
Modelo
API
Agregación
Fuentes
Solicitudes
UI Solicitides
Protocolo MetaMapa
Fuente De Datos Estática
Principal
API
Fuentes
Dataset
Fuente De Datos Proxy
Modelo
API
Fuentes
Servicio Externo
Servicio de estadísticas
Modelo
Estadísticas
Generador de Estadísticas
Fuentes
UI Visualizador
Visitante
Fuente De Datos Dinámica
API
Modelo
Fuentes
Hechos y solicitudes
UI Editor
Solicitudes
UI Solicitides
OAuth
Proveedor de Identidad
Visitante
Revisor