UCLM
UNIVERSIDAD DE CASTILLA-LA MANCHA

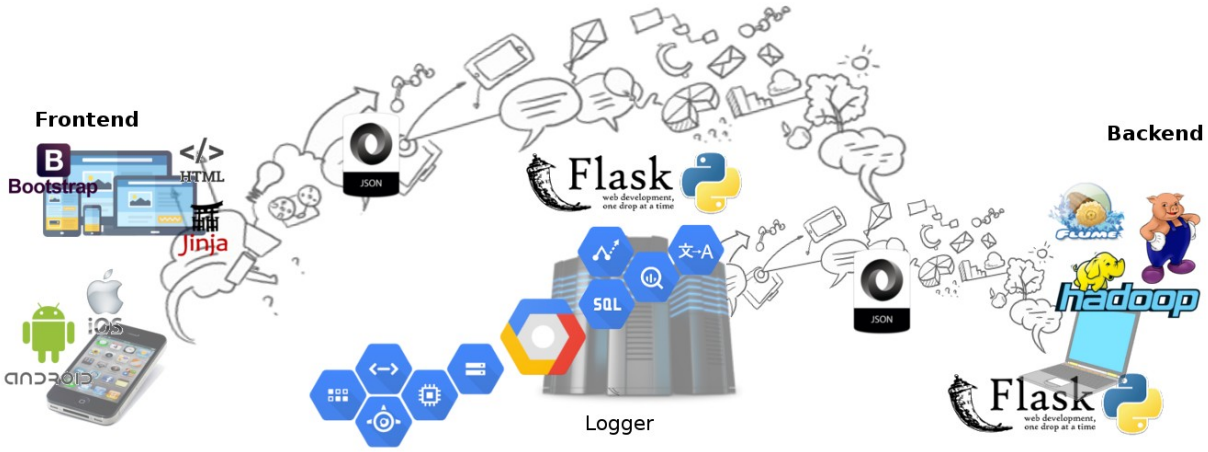
Escuela Superior de Informática

Graduado en Ingeniería Informática
Aplicaciones Distribuidas en Internet

– Práctica 2017/18: Twitter –

Puntuación (nivel básico): 18
Puntuación por objetivos/tareas: 32
Fecha límite de entrega: TBA (Enero 2018 – Necesaria defensa)
Método de trabajo: Individual o por parejas

La práctica se divide en tres partes bien diferenciadas: un backend, un logger y un frontend. El backend realizará las tareas más costosas en términos computacionales y correrá sobre el PC del alumno. El servidor logger será un servicio web y estará alojado en Google. Por último, el frontend será un cliente web o app móvil que permitirá al usuario interaccionar con el sistema creado. En la siguiente imagen se muestra una visión general del sistema a construir. Todas las comunicaciones se realizarán mediante una codificación JSON.



Backend *(8 puntos)*

El Backend albergará el procesamiento de la aplicación, en ella se desplegará Hadoop junto con los servicios Pig y Flume como se han visto en clase. Flume estará conectado a Twitter con el fin de rescatar una serie de tweets, los cuales serán analizados mediante Pig. La obtención de tweets con Flume puede ser sustituida cargando los ficheros con tweets en el cluster manualmente, pero ello supondrá una reducción en la nota final (ver rúbrica). Entre el cluster y el resto de la práctica se albergará un servicio web capaz de obtener los tweets y analizarlos, este servicio web será implementado en Flask con las siguientes operaciones:

- Inicio de la adquisición de tweets. Este método iniciará Flume con la intención de obtener una colección de tweets. Se pueden filtrar utilizando keywords en la configuración de Flume. En el caso de elegir la inserción manual este punto no es posible implementarlo.
- Parada de adquisición de tweets. Este método para la ejecución de la operación anterior. En el caso de elegir la inserción manual este punto no es posible implementarlo.

- Servicio de análisis. Este servicio ofrecerá la posibilidad de analizar mediante Pig los tweets recolectados. Puede ser dividido en varios sub-servicios de forma que implemente al menos uno de los siguientes (cada servicio adicional implementado supondrá 3 puntos adicionales)
    - Filtra los tweets por su idioma: ingles o español.
    - Lista aquellos que contengan una determinada palabra.
    - Muestra aquellos que han sido retweeteados por Z personas.
    - Muestra los tweets con más de Y likes.

Las operaciones pueden ser hechas ad-hoc o proporcionar al usuario la posibilidad de modificar los parámetros. Por ejemplo Para la última operación, el desarrollador podría fijar el valor Y, o por el contrario, proveer al usuario de los mecanismos necesarios para que éste indique el valor de Y (ver rúbrica para la puntuación).

Puntos adicionales en el Backend (18 puntos):

- Implementar más de un servicio de análisis (3 ptos por cada uno)
- Desplegar Hadoop, Pig y Flume mediante virtualización. Se podrá utilizar Docker o Vagrant y deberá contener al menos dos nodos esclavos (4 puntos, similar a la tarea 3).
- Implementar un sistema de autenticación de forma que los usuarios autorizados sean los únicos que puedan utilizar el sistema. (1 punto HTTPAuthBasic, 4 puntos OAuth).
- Utilización de certificados (1 punto)

Logger *(6 puntos)*

El servidor logger contendrá un registro de las operaciones realizadas por parte de los usuarios, es decir actuará de logger de toda la actividad realizada en nuestro servicio. Para ello se implementará de nuevo un servicio web en Flask que estará albergado en Google y registrará la operación realizada, el usuario que la realizó (si es posible), la fecha y la hora. También permitirá rescatar toda la información almacenada de la siguiente forma:

- Todos los logs
- Filtrado por un intervalo de fechas

Puntos adicionales en el Logger (4 puntos)

- Utilizar el Datastore de Google (2 puntos)
- Utilizar Memcache (2 puntos)

Frontend *(4 puntos)*

El frontend mostrará una interfaz funcional (no es necesario "disfrazarla" o "decorarla") que permita interaccionar con el backend de nuestra aplicación. En ella se deberá mostrar al menos las operaciones de captura y parada de Tweets y una de las operaciones implementadas en el backend, junto a la entrada de datos necesarios para dicha operación si tal es el caso (recordad que puede hacerse a medida).

Otros puntos adicionales (8 puntos)

- Realizar tests a los servicios webs implementados en Flask (6 puntos)
- Utilizar webhooks de forma que el backend informe al frontend que la operación ha finalizado, y éste muestre un mensaje flashing (4 puntos)

Modalidades para el desarrollo de la práctica

Si se ha elegido la modalidad por ***parejas***, éstas deberán utilizar un repositorio (GitHub o Bitbucket) para ver las actividad de cada participante. Además, se recomienda el uso de algún sistema que permita asignar responsabilidades o tareas (Trello). Para el caso ***individual*** también es recomendable el uso de repositorios.

Recomendaciones

Se recomienda enérgicamente el uso de repositorios para el desarrollo de la práctica, así como el uso de un *framework* de tests con el fin de probar aquello que ha sido implementado.

El primer paso será definir las interfaces API REST de nuestro servicio de backend y logger que vamos a ofrecer, para ello es conveniente asociar las operaciones HTTP de una determinada URL a uno de los métodos/operaciones permitidas. Seguidamente, se pasará la solución obtenida a Python mediante Flask, donde las funciones que manejan los endpoints no contendrán funcionalidad alguna, es decir lo que se crea es la estructura que tendrá el servicio web.

A la vez o en un segundo paso, se realizarán los tests correspondientes a cada uno de los *endpoints* diseñados en el paso anterior, comprobando que los códigos de estado devueltos son los correctos.

Posteriormente se implementará la lógica necesaria de cada endpoint junto con su test correspondiente. Seguidamente, se tomará la decisión sobre qué puntos adicionales serán abordados.

Referencias:

Flask – flask.pocoo.org
Jinja2 – jinja.pocoo.org
Werkzeug – werlzeug.pocoo.org
JSON – json.org
Schema – schema.org
JSON-LD – json-ld.org
GCP – https://cloud.google.com/