**Escenario y Pruebas de Estrés API REST y Batch**

**Integrantes**

* **Camilo Ramírez Restrepo**   
  **Leidy Viviana Osorio Jiménez**
* **Laura Daniela Molina Villar**
* **Tim Ulf Pambor**
* **Shadith Yiseli Pérez Rivera**

**Objetivo**

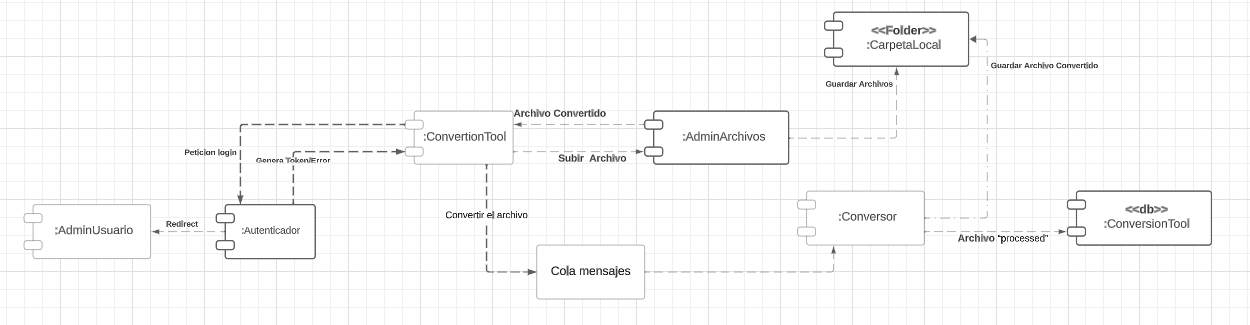
El objetivo de este plan es evaluar la capacidad de la aplicación de convertidor de archivos y su infraestructura de soporte en un entorno tradicional para determinar sus máximos aceptables. El objetivo es comprender cómo la aplicación responde a diferentes niveles de carga de usuarios y cuál es su capacidad máxima.

**Objetivos específicos**

* Medir la capacidad de procesamiento de la aplicación en términos de transacciones por minuto.
* Evaluar el tiempo de respuesta promedio de la aplicación bajo diferentes cargas de usuarios.
* Analizar la utilización de recursos, como CPU, memoria y red, durante las pruebas de carga.
* Identificar cuellos de botella y posibles áreas de mejora en la infraestructura y la aplicación.

**Descripción general**

Aplicación web que ofrece gratuitamente a usuarios de internet, puedan subir abiertamente diferentes formatos multimedia de archivos y cambiarles su formato o realizar procesos de compresión. El modelo general de funcionamiento de la aplicación se basa en crear una cuenta en el portal web y acceder al administrador de archivos. Una vez la cuenta ha sido creada, el usuario puede subir archivos y solicitar el cambio de formato de estos para descargarlos. La aplicación web le permite a un usuario convertir archivos multimedia en línea de un formato a otro, seleccionando únicamente el formato destino.



**Tipos de pruebas a realizar**

* **Pruebas de carga:**

Las pruebas de carga se realizan para evaluar cómo se comporta la aplicación bajo una carga de usuarios típica. Se simulan múltiples usuarios simultáneos realizando acciones en la aplicación, como navegación, búsqueda, compras, etc. El objetivo es medir el tiempo de respuesta promedio y asegurarse de que la aplicación funcione dentro de los límites aceptables bajo una carga normal.

* **Pruebas de estrés:**   
  Las pruebas de estrés se utilizan para determinar el punto de quiebre de la aplicación y cómo responde bajo una carga excesiva. Se aumenta gradualmente la carga de usuarios hasta que se alcanza o supera el límite de capacidad de la aplicación. El objetivo es identificar cómo se comporta la aplicación bajo condiciones extremas y si es capaz de recuperarse sin fallos graves.

**Configuración del sistema**

* **Aplicación Flask**
* **Servidores EC2:**
  + Tipo de Instancia: Se utilizarán instancias de tipo c5.xlarge,
  + Capacidad de CPU: Cada instancia c5.xlarge proporciona 4 vCPUs, lo que permite ejecutar múltiples instancias en paralelo para simular usuarios concurrentes.
  + Sistema Operativo: Se utilizará el sistema operativo Linux (por ejemplo, Amazon Linux 2) en las instancias EC2 para ejecutar las pruebas.
  + Configuración de Red: Las instancias EC2 estarán configuradas en una red privada virtual (VPC) de AWS.
  + Monitoreo: Este monitoreo se realiza con Google Cloud Monitoring con un Agente de operaciones en los servidores GCP, donde estará alojada la aplicación.

**Escenarios**

Se han identificado dos escenarios claves.

* **Escenario 1 – API REST de agregar una tarea**

**Descripción de funcionalidad**

Agregar tareas a la cola de mensajería cuando se solicite la conversión de un video

**Objetivo**:Cuando se demora en enviar la petición a la cola de mensajería

**Métricas**

* Throughput: cantidad de conversiones realizadas por minuto.
* Tiempo de respuesta promedio: tiempo que tarda la aplicación en procesar una conversión.
* Utilización de recursos: monitoreo de la CPU, memoria y uso de red durante las pruebas.

**Herramientas para la prueba**

Se utilizará JMeter como la herramienta principal para realizar las pruebas de capacidad. se realizarán 1000 peticiones con grupos concurrentes de 100 hasta llegar al tope.

**Criterios de aceptación**

* **Tiempo de Respuesta:**

El tiempo de respuesta promedio de la aplicación en todos los escenarios de prueba no debe superar los 0.5 segundos en el 95% de las transacciones, por petición.

El tiempo de respuesta en ningún escenario de prueba debe superar los 4 segundos en el 99% de las transacciones.

**Datos de prueba**

Se utilizarán archivos multimedia de ejemplo que representen escenarios reales de uso. Estos archivos deben ser representativos de los diferentes formatos admitidos por la aplicación video.

**Iteraciones**

* Se realizarán máximo 10 iteraciones o la cantidad en la cual se cumpla el tope de la aplicación.

**Riesgos**

* Sobrecargar el entorno de producción, lo que podría resultar en una degradación del servicio o incluso una interrupción.
* Aprendizaje de la herramienta
* **Escenario 2 - Conversión de videos**

**Descripción de funcionalidad**

La funcionalidad implica la conversión de videos en formato MP4 a webm, esta conversión es necesaria para adaptar videos a diferentes requerimientos de visualización.

**Métricas**

* + **Requests per second**: Cantidad de peticiones por minuto en la conversión de videos. El sistema debe ser capaz de manejar un mínimo de 100 solicitudes por segundo para la conversión de videos MP4 a Webm
  + **Time per Request** : Tiempo que tarda la aplicación en procesar las peticiones. El tiempo promedio por solicitud de conversión no debe exceder los 5 segundos.
  + **Time per Request:** Tiempo promedio en atender una petición. El tiempo de conversión por petición no debe exceder lo 10 segundos en más del 5% de las solicitudes.

**Herramientas para la prueba**

* **Conversor de video:**
* **Script Python**: Se creará un script en Python que envíe datos a la cola de mensajería, simulando múltiples solicitudes concurrentes al sistema de conversión de videos.

**Criterios de aceptación**

* **Tiempo de Respuesta:**
  + El sistema debe ser capaz de manejar un mínimo de 100 solicitudes por segundo para la conversión de videos MP4 a Webm
  + El tiempo promedio por solicitud de conversión no debe exceder los 5 segundos.
  + El tiempo de conversión por petición no debe exceder lo 10 segundos en más del 5% de las solicitudes.

**Datos de prueba**

* Se utilizarán archivos multimedia de ejemplo que representen escenarios reales de uso. Estos archivos deben ser representativos de los diferentes formatos admitidos por la aplicación video.

**Iteraciones**

* + Se realizarán máximo 10 iteraciones o la cantidad en la cual se cumpla el tope de la aplicación.

**Riesgos**

* + Sobrecargar el entorno de producción, lo que podría resultar en una degradación del servicio o incluso una interrupción.
  + Aprendizaje de la herramienta
  + Este escenario se debe realizar mediante un script