# Entrega 4 – Sistema de Conversión Cloud

Daniel Andrés Jiménez Riveros
Andrés Martín Ochoa Toro
Esteban Emmanuel Ortiz Morales
Manuel Felipe Porras Tascón
Desarrollo de Soluciones Cloud
Universidad de los Andes, Bogotá, Colombia

Enlace del Repositorio:
https://github.com/AmOchoat/Cloud-Entrega-1

Enlace Documentación Postman:

https://documenter.getpostman.com/view/5689272/2s93CPrYBi

**Contexto del problema**:

Una nueva compañía de cloud denominada *Cloud Conversion Tool* desea crear una aplicación web que será ofrecida gratuitamente a usuarios de internet para que estos 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.

En esta primera versión, la herramienta sólo soporta la compresión de archivos en con tres tipos diferentes de algoritmos y utilidades, estos algoritmos son ZIP, TAR.GZ y TAR.BZ2.

**Implementación de la solución**:

Para brindar una solución a la compañía *Cloud Conversion Tool,* se realizó la implementación del API REST que permite la creación de una cuenta (*sign up)*, el inicio de sesión(*login*), la creación de una nueva tarea de compresión, el listado de todas las tareas de compresión de un usuario, borrar una tarea de compresión de un usuario y la obtención del archivo original o procesado.

Así mismo se realiza la implementación de una interfaz de usuario que permita consumir el API REST mencionada previamente.

**Arquitectura realizada:**

A partir de las necesidades manifestadas por el negocio, se definió el siguiente conjunto de diagramas conformado por el diagrama de clases, diagrama de componentes y diagrama de despliegue.

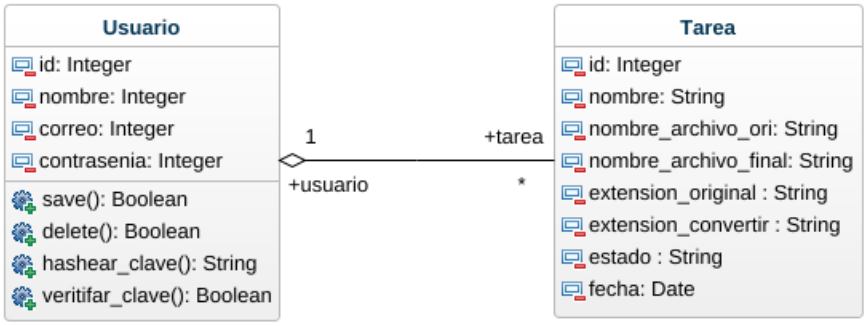


Imagen 1. Diagrama de clases entrega 3 *Cloud Conversion Tool*

En este diagrama se define el usuario y la tarea de conversión. Un usuario tiene muchas tareas de conversión, donde cada tarea tiene un nombre y nombre de los archivos generados, así como la extensión de inicio y final. Lo más importante en este modelo es el estado de una tarea, que puede ser *uploaded* y *processed,* y los nombres de los archivos. Para identificar un archivo en caso de ser necesario, se toma el nombre del archivo asociado a la tarea como valor de texto y se busca este nombre en un directorio de archivos internos.

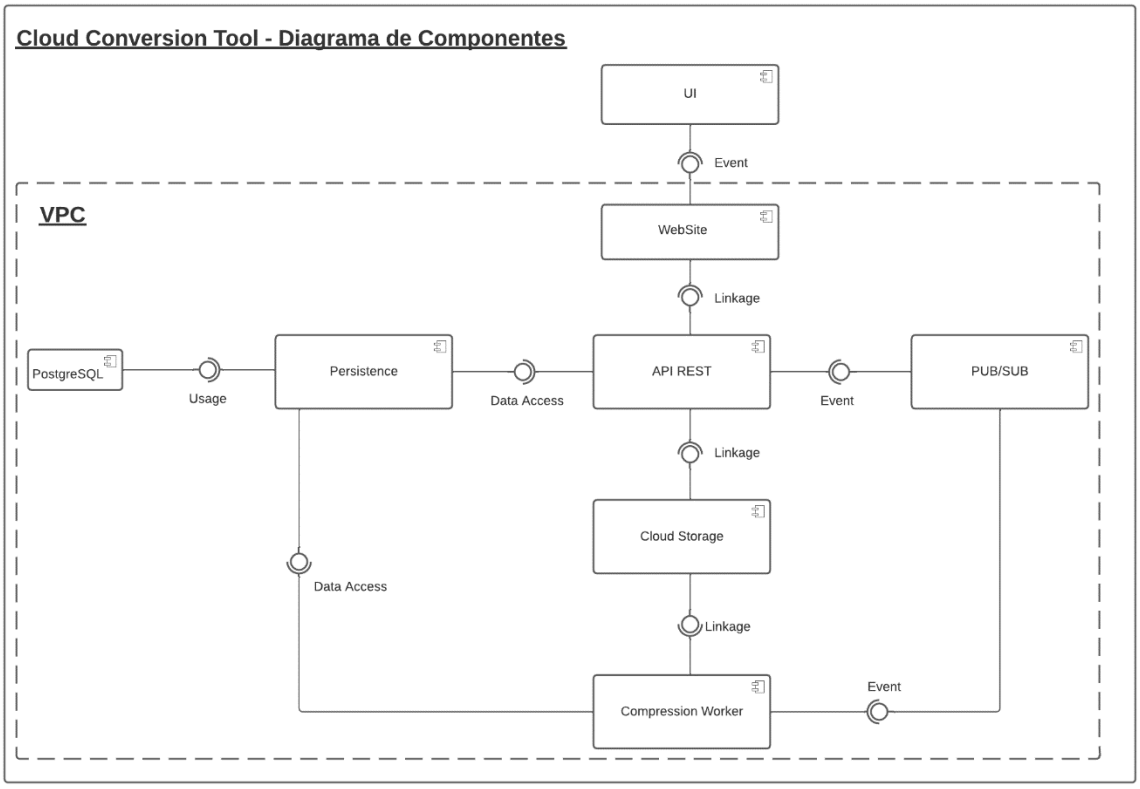


Imagen 2. Diagrama de Componentes entrega 3 *Cloud Conversion Tool*

Nuestros componentes principales son una VPC que envuelve los otros componentes. Dentro de la VPC está el Front-end de la página web construido en Reactjs el cual está conectado al Back-end construido en Flask. Dentro de la VPC también se encuentra el API REST que tiene como objetivo conectar el backend con el frontend, además de tener a un componente Cloud storage para manejar los temas de almacenamiento de los archivos convertidos junto con los componentes PUB/SUB y Compression Worker que son los encargados de controlar todo el flujo de conversión de archivos de la aplicación.

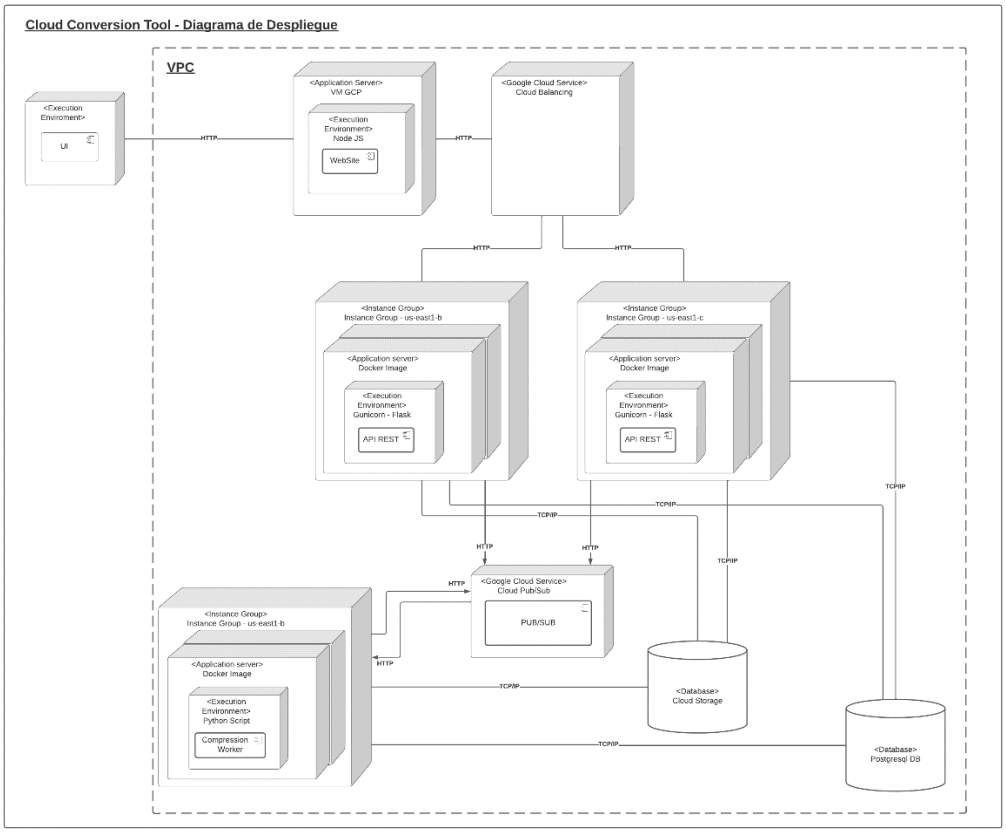


Imagen 3. Diagrama de Despliegue entrega 4 *Cloud Conversion Tool*

En el caso de los diagramas de componentes y diagrama de despliegue se observa la relación de los elementos implementados y los servicios usados en su despliegue. Para el desarrollo de los componentes se hace uso de los servicios requeridos en el curso. En este caso se usa *Flask* para el desarrollo del API, *PUB/sub de Google cloud para la conversión de archivos, un Balanceador de cargas para el balanceo de las peticiones que se realizan para convertir archivos. Además, se tiene que Cloud Storage para el almacenamiento de los archivos que se suben y se convierten en la aplicación. Finalmente, estas dos instancias se encuentran conectadas a la misma base de datos para mantener la integridad de la información de la aplicación y están conectadas también al PUB/SUB que es el encargado de conectarse al compression worker para poder realizar la conversión de los archivos.*

Así pues, para esta entrega se creó un VPC con el fin de añadirle una capa de seguridad al Back-End y a la base de datos. Esto con el fin que demás usuarios no

puedan acceder directamente a estos componentes. Dentro de la VPC se creó una máquina virtual encargada de manejar el Front-End de la aplicación y este se conecta con un balanceador de cargas que maneja la carga de las peticiones realizadas direccionándolas a dos posibles grupos de distancias las cuales cada una de ellas se encuentra en una zona diferente.