Instituto Tecnológico de Costa Rica

Área académica ingeniería en computadores

Soa41d: Arquitectura Orientada A Servicios Aplicada A Sistemas Emergentes

TÍTULO:
“Documentación del Rol de Arquitecto de Infraestructura”

AUTORES:
Allan Calderón Quirós
Kevin Acevedo Rodríguez
Pablo Carmona Ulate
Nickolas Rodríguez Cordero

Cartago, Costa Rica
2022

Justificación del diseño

El sistema que se debe diseñar debe permitir una correcta administración de gastos mensuales de departamentos individuales. En ese sentido los empleados ejecutivos de la empresa son los encargados de consumir la información; el resto de empleados son los encargados de ingresar los datos de cada departamento. Es importante que la información siempre se mantenga en la base de datos, sin embargo, solo debe mostrar en la UI la información correspondiente del mes actual.

De este modo el sistema en cuanto a interfaz de usuario debe estar dividido en dos partes, una debe ser capaz de guardar todos los gastos ingresados por los usuarios por medio de una página web. La otra parte debe ser también una página web que muestre todos los datos del mes correspondiente junto con los 3 departamentos que más han gastado y el gasto total mensual.

Se requería que el sistema estuviera basado en los principios DDD con el fin de facilitar la comunicación, mejorar la flexibilidad y darle una mayor prioridad al dominio. Se explica la aplicación de los tres principios de DDD a continuación:

- Centrarse en el dominio principal y en la lógica del dominio: Se definieron las partes principales del dominio: las dos interfaces gráficas revisión de datos, ingreso de datos y fuente de datos.
- Basar los diseños complejos en modelos del dominio: El diseño de la arquitectura se hizo basado en la terminología anteriormente mencionada, sólo se agregó la interacción con la base de datos y el load balancer.
- Colaborar constantemente con los expertos del dominio, con el fin de mejorar el modelo de aplicación y resolver cualquier problema emergente relacionado con el dominio: El equipo de trabajo se reunió de diferentes ocasiones para ir modificando la terminología a utilizar y depurarla hasta que el resultado final fuera satisfactorio y reflejara en su totalidad lo que se quería expresar, sin ambigüedades.

Diseño inicial de la arquitectura (Arquitectura Prescriptiva)

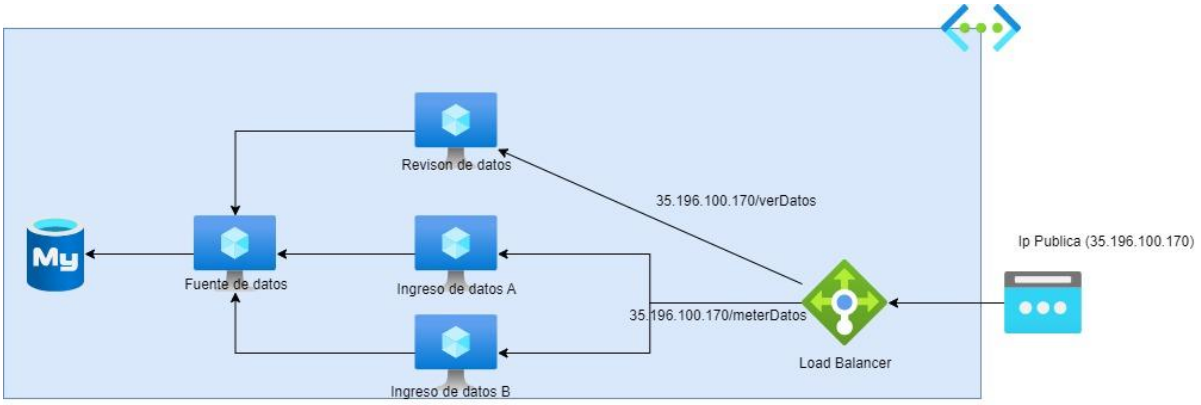
Para la arquitectura prescriptiva se tomó la decisión de utilizar 4 máquinas virtuales. La primera de ellas se utiliza para correr el servicio de fuente de datos. La segunda máquina virtual se encarga de contener el servicio de visualización de datos y finalmente las otras 2 máquinas virtuales restantes; cada una contiene una instancia del servicio de introducción de datos.
Como se puede ver en la siguiente imágen, se tiene un balanceador de carga mediante el cual se accede a la aplicación y sus diferentes servicios ya que este balanceador de cargas es el único acceso y la manera de alcanzar el servicio de visualización de datos por ejemplo es utilizando el ruteo de dominos

A continuación se describe la conexión que está asociada a cada uno de los servicios mostrados en la imágen:

- **Fuente de datos:** Este servicio es el único que tiene comunicación con la base de datos (MySQL) y se encarga de proveer una REST API para poder ingresar nuevos gastos a la base de datos y también para poder obtener información con los gastos del mes.

- **Mostrar Datos:** Este servicio consiste de una aplicación web que se encarga de consultar a la fuente de datos la información de los gastos del mes y mostrarla en pantalla.

- **Ingresar Datos:** Este servicio consiste en una aplicación web que permite a los usuarios (empleados de la empresa) registrar gastos realizados. Esta información es enviada a la fuente de datos para luego ser almacenada en la base de datos.
- **My:** El servicio My consiste en una base de datos que utiliza el motor de MySQL. En esta base de datos se guardan los diferentes reportes de gastos que genera cada empleado atravez del portal de “Ingresar Datos” y estos pueden ser visualizados usando el portal de “Mostrar Datos”, ambos servicios se comunican con el servicio de “Fuente de Datos”.

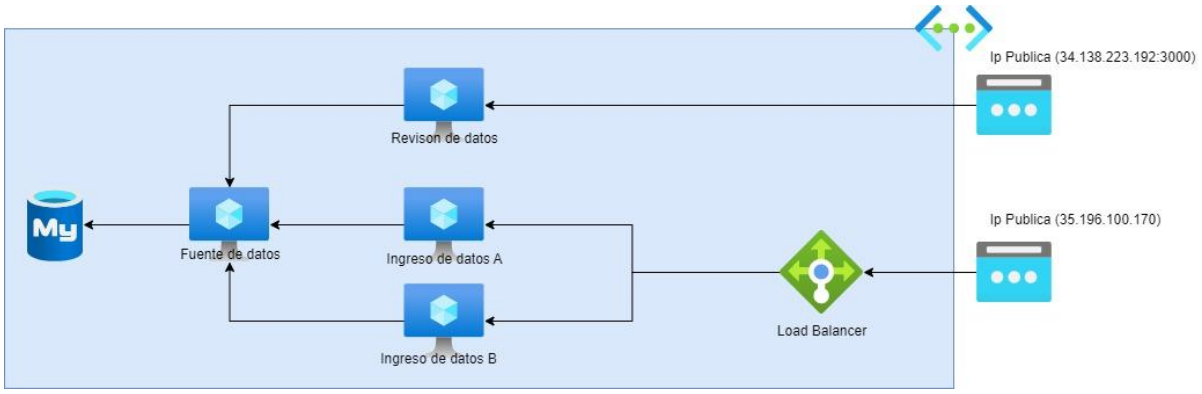


Diseño final de la arquitectura (Arquitectura Descriptiva)

Para el diseño de la arquitectura final se realizaron algunos cambios necesarios para que el sistema pueda funcionar:

1. Se decidió excluir el servicio de revisión de datos del flujo del balanceador de cargas debido a que no era posible el ruteo de solicitudes sin tener un nombre de dominio y la razón por la que no se optó por la adquisición de un dominio fue porque se salía de presupuesto del proyecto.

Los demás servicios no necesitaron cambio alguno ya que su papel es simplemente el procesamiento de los datos y la transferencia de datos hacia la fuente de datos el cual es el único con acceso a la base de datos.



Descripción de todos los servicio

Cada uno de los servicios fué trabajado localmente y dentro de un repositorio de GitHub. Luego de realizar las pruebas respectivas para cada una de las partes del sistema, se crearon máquinas virtuales en la plataforma de Google Cloud y luego se le integró el código fuente dentro de las VMs para crear los servicios.
A continuación se presenta una breve descripción de cada uno de los servicios involucrados en el proyecto.

Servicio	**Lenguaje**	**Descripción**
fuente de datos	Javascript (Express JS)	Este servicio es el único que tiene comunicación con la base de datos (MySQL) y se encarga de proveer una REST API para poder ingresar nuevos gastos a la base de datos y también para poder obtener información con los gastos del mes.
Mostrar Datos	Javascript (React JS)	Este servicio consiste de una aplicación web que se encarga de consultar a la fuente de datos la información de los gastos del mes y mostrarla en pantalla.
Ingresar Datos	Python (Flask)	Este servicio consiste en una aplicación web que permite a los usuarios (empleados de la empresa) registrar gastos realizados. Esta información es enviada a la fuente de datos para luego ser almacenada en la base de datos.

Funcionamiento del Balanceador de Carga

El balanceador de carga se asegura de que el tráfico web no se concentre en un solo servidor, el cual eventualmente terminaría saturando debido a las miles de miles de peticiones por segundo generadas por los clientes.
Su objetivo lo cumple direccionando a los clientes hacia el servidor que cuente con la mayor disponibilidad y el mismo contenido que estos solicitan. Es un proceso completamente transparente para quien accede a determinado sitio, por lo que a simple vista no podemos detectar esto.

Modelo de despliegue

Para este proyecto se tomó la decisión de hacer el despliegue de los servicios mediante el modelo “Un servicio por instancia de VM”.
La razón es que, para cumplir con un principio de responsabilidad única en cada uno de los servicios y además de que, en caso de realizar un cambio en alguna de las partes del sistema, no se tenga que detener o modificar los demás componentes.
Otra razón es que cada uno de los servicios propuestos debía estar expuesto en un puerto y dirección IP en específico (En este caso, estos parámetros son dados por cada instancia de máquina virtual).