Pedidos de comida

Infraestructura como código

PONDERACIÓN: 5

Horas Aproximadas: 30

Universidad San Carlos de Guatemala

Facultad de ingeniería.

Ingeniería en ciencias y sistemas

Índice

Competencias

Aplica y adapta modelos de despliegue en la nube mediante arquitecturas por capas, microservicios y plataformas como Docker y Terraform optimizando la disponibilidad y escalabilidad de los sistemas.

Objetivos

General

Diseñar, construir y desplegar automáticamente una plataforma web de gestión de un menú de comida y pedidos, utilizando una arquitectura de microservicios distribuida en la nube, y garantizando la alta disponibilidad y escalabilidad del sistema mediante el uso de Infraestructura como Código (IaC) con Terraform y gestión de configuración con Ansible.

Específicos

- Diseñar y modelar el patrón arquitectónico de microservicios clave (ej., Servicio de Menú, Servicio de Pedidos, Servicio de Usuarios) sobre una arquitectura de capas, definiendo la API Gateway como punto de entrada y especificando los contratos de las APIs.
- Establecer el modelo de persistencia para cada microservicio, aprovisionando una base de datos desacoplada y escalable (ej., una base de datos gestionada o *managed database*) usando Terraform.
- Contenerizar todos los microservicios y el *frontend* usando Docker y definir un proceso de *build* e *push* a un Registro de Contenedores en la nube.
- Aprovisionar toda la infraestructura de la nube usando Terraform (incluyendo redes, máquinas virtuales o clúster de contenedores, y un Balanceador de Carga) para garantizar el aislamiento y la alta disponibilidad de los recursos.
- Automatizar la configuración del sistema operativo y el despliegue de los contenedores usando Ansible, garantizando que los servicios se lancen con las configuraciones de red y seguridad adecuadas para integrarse correctamente con el Balanceador de Carga y las bases de datos.

Descripción / Enunciado

El objetivo del proyecto es desarrollar una **plataforma web** simple que permita a un restaurante gestionar y visualizar su **menú de productos** en tiempo real. Los usuarios (camareros o clientes) podrán consultar el listado de categorías, los productos disponibles, sus descripciones y precios.

Para alcanzar este objetivo, se implementará **un patrón arquitectónico de microservicios** de dos capas, con responsabilidades claras:

- **Microservicio de Productos:** Encargado de gestionar los datos de los productos individuales (nombre, descripción, precio, estado).
- **Microservicio de Categorías:** Encargado de gestionar las agrupaciones del menú (ej. Entradas, Platos Fuertes, Postres) y enlazar los productos a ellas.

Ambos microservicios simularán el acceso a sus respectivos datos maestros a través de *endpoints* dedicados.

El **Frontend** de la plataforma consumirá los servicios de ambos microservicios para construir y mostrar el menú completo. Toda la infraestructura se desplegará de forma totalmente automatizada utilizando **Terraform** y **Ansible** en una **Plataforma en la Nube** (AWS o Google Cloud Platform).

Requisitos Clave: IaC y Despliegue

Infraestructura como Código con Terraform: Se utilizará **Terraform** para definir y aprovisionar la infraestructura base. Esto incluye la reserva de **direcciones IP públicas estáticas** para las máquinas virtuales que alojarán el frontend y los dos microservicios. La asignación estática facilita el acceso y las pruebas consistentes en la nube. Además, Terraform se encargará de establecer el modelo de persistencia para cada microservicio mediante el aprovisionamiento de bases de datos gestionadas (*managed databases*) que sean desacopladas y escalables. Esto asegura que la infraestructura completa (computación y datos) sea reproducible, inmutable y esté definida enteramente como código.

Contenerización y Automatización con Docker y Ansible: Todos los componentes (frontend, Microservicio de Productos y Microservicio de Categorías) deberán ser **contenedorizados** usando **Docker**.

Se empleará **Ansible** para automatizar la etapa de configuración (*provisioning* post-creación) y el despliegue final. Esta automatización no solo incluirá la instalación de Docker Engine, sino también la **configuración del sistema operativo** de las máquinas virtuales. Ansible será el responsable de lanzar los contenedores con las **configuraciones de red y seguridad adecuadas**, lo cual es crucial para su correcta integración tanto con el **Balanceador de Carga** (para el tráfico entrante) como con las **bases de datos gestionadas** (para la persistencia). Este proceso asegura que la arquitectura sea completamente portable y se pueda recrear de forma idéntica en cualquier entorno.

Documentación Solicitada

La documentación del proyecto deberá incluir:

1. **Diagrama de Arquitectura de Despliegue:** Un diagrama que muestre la interacción entre el frontend, los dos microservicios y la forma en que **Terraform/Ansible** gestionan el despliegue en la nube.
2. **Manual de usuario:** Un documento que detalle los pasos para interactuar con la plataforma web (frontend). Debe incluir instrucciones claras sobre cómo visualizar el menú, cómo navegar entre categorías y cómo consultar los detalles de los productos, dirigido al usuario final (camareros o administradores del restaurante).

Entregables

- Link a repositorio privado en github a través UEDI.

Consideraciones

- Todos los integrantes deben estar presentes el día de la calificación, ya que se validará la participación de los integrantes.
- Agregar al aux de su sección correspondiente desde el primer día después de la publicación del enunciado:

    - A: mmynor97
    - B: MelyzaRC
- La documentación debe ser almacenada en el repositorio utilizando el formato MarkDown
- **Las copias parciales o totales tendrán una nota de 0 y será reportada a la escuela de sistemas.**
- La calificación obtenida puede cambiar si la ingeniera solicita una revisión.
- Todos los integrantes del grupo deberán tener commits sustanciales a lo largo del desarrollo. Utilizar el formato de commits para dejar prueba de su trabajo. Hacer merge de pull requests no cuenta como commit válido.
- El nombre del repositorio debe seguir el formato AYD2_<seccion>_2S2025_PRAC2_G<#>(eg.,AYD2_A_2S2025_PRAC2_G4)

Cronograma

Tarea	Fecha
Asignación de la práctica / Entrega del enunciado	16 Octubre 2025
Fecha de entrega	23 Octubre 2025
Fecha de calificación	25 Octubre 2025

Valores

En el desarrollo de la práctica, se espera que cada estudiante demuestre honestidad académica y profesionalismo. Por lo tanto, se establecen los siguientes principios:

1. **Originalidad del Trabajo**
    - Cada estudiante o equipo debe desarrollar su propio código y/o documentación, aplicando los conocimientos adquiridos en el curso.
2. **Prohibición de Copias y Plagio**
    - Si se detecta la copia total o parcial del código, documentación o cualquier otro entregable, la calificación será de **0 puntos**.

- Esto incluye la reproducción de código entre compañeros, la reutilización de proyectos de semestres anteriores o el uso de código externo sin la debida referencia.

3. **Uso Responsable de Recursos Externos**
    - El uso de bibliotecas, frameworks y ejemplos de código externos está permitido, siempre y cuando se referencian correctamente y se comprendan plenamente. (Consultar con el catedrático su política)
4. **Revisión y Detección de Plagio**
    - Se podrán utilizar herramientas automatizadas y revisiones manuales para identificar similitudes en los proyectos.
    - En caso de sospecha, el estudiante deberá justificar su código y demostrar su desarrollo individual o en equipo. Si este extremo no es comprobable la calificación será de **0 puntos**.

Al detectarse estos aspectos se informará al catedrático del curso quien realizará las acciones que considere oportunas.