Universidad San Carlos de Guatemala
Facultad de ingeniería.
Ingeniería en ciencias y sistemas

Título del Proyecto:

Delivereats

PONDERACIÓN: 20

Horas Aproximadas: 45

Índice

Contenido

Contenido

1. Resumen del proyecto

El proyecto consiste en el desarrollo de una plataforma de entrega de alimentos tipo *delivery*, diseñada bajo una arquitectura de microservicios. La aplicación permitirá a los usuarios registrarse, iniciar sesión, explorar restaurantes, consultar menús y generar pedidos. Asimismo, incluirá un módulo básico de gestión para repartidores y propietarios de restaurantes.

2. Competencias que desarrollaremos

- Capacidad para diseñar, estructurar y desplegar servicios independientes que trabajen de forma coordinada.
- Implementación de APIs REST para consumo externo y comunicación interna basada en gRPC.
- Uso de JWT, manejo de roles y protección de endpoints sensibles.
- Diseño de modelos y gestión de bases de datos relacionales para usuarios, restaurantes, menús y pedidos.
- Diseño de componentes flexibles que permitan agregar nuevas funcionalidades en fases posteriores.

3. Objetivos del Aprendizaje

3.1 Objetivo General

Aplicar los conocimientos adquiridos a lo largo de la carrera para desarrollar una aplicación basada en arquitectura de microservicios, integrando principios modernos de ingeniería de software como autenticación segura, persistencia de datos, comunicación entre servicios y diseño escalable, con el fin de consolidar competencias técnicas fundamentales en el desarrollo de sistemas distribuidos.

3.2 Objetivos Específicos

- Implementar un servicio de autenticación que gestione el registro, inicio de sesión y control de roles utilizando JWT.
- Diseñar e implementar microservicios independientes para la administración de restaurantes, menús y pedidos, cada uno con su propia base de datos.
- Configurar un API Gateway que centralice el acceso externo y gestione las solicitudes hacia los microservicios internos.
- Establecer la comunicación entre servicios mediante REST y gRPC, definiendo contratos claros y desacoplados.
- Documentar la arquitectura, los componentes, los endpoints y las decisiones técnicas realizadas para garantizar claridad y mantenibilidad del proyecto.
- Uso de Docker para creación de imágenes y Docker-Compose para levantar servicios
- Uso de GCP para despliegue de la aplicación

4. Enunciado del Proyecto

4.1 Descripción del problema a resolver

Pequeños comercios y emprendedores suelen manejar pedidos, inventarios y entregas mediante mensajes informales, cuadernos, llamadas o redes sociales. Esta falta de un sistema unificado provoca errores en los pedidos, información perdida, registros incompletos y mala coordinación entre clientes, proveedores y transportistas.

La ausencia de una plataforma moderna también limita el crecimiento, ya que los dueños no pueden automatizar procesos ni agregar nuevas funciones sin rehacer completamente sus herramientas actuales. Esto genera cuellos de botella y dificulta escalar su operación.

Para resolver esta situación se plantea el desarrollo de una plataforma distribuida que permita gestionar usuarios, manejar pedidos, registrar productos o servicios e integrar módulos adicionales en fases posteriores. La arquitectura de microservicios permitirá construir un sistema flexible y seguro, con servicios independientes que se comuniquen entre sí mediante REST y gRPC, aplicando buenas prácticas de diseño, autenticación con JWT y persistencia de datos.

4.2 Alcance del proyecto

4.2.1 Microservicios mínimos:

1. API Gateway / BFF
2. Auth-Service
3. Restuarant-Catalog-Service
4. Order-Service
5. Delivery-Service
6. Notification Service

API Gateway (REST)

- Expone las rutas **REST** al frontend
- Validar JWT
- Autorización básica por roles
- Enrutar a otros servicios GRPC

Auth-Service:

El servicio de Autenticación y Usuarios (Auth-Service) es el responsable de la gestión de usuarios y del manejo de tokens JWT para el acceso seguro al resto de microservicios de la plataforma. Es utilizado por todos los roles del sistema: CLIENTE, RESTAURANTE, REPARTIDOR y ADMINISTRADOR.

Este servicio define un **modelo mínimo de usuario**, que incluye:

- **Id**: Identificador único del usuario dentro del sistema.
- **Email**: Correo electrónico del usuario, utilizado como credencial principal para el inicio de sesión.
- **Contraseña (encriptada)**: Clave de acceso almacenada siempre de forma segura (encriptada/hasheada) para proteger la información del usuario.
- **Rol**: Define el tipo de usuario dentro de la plataforma (**CLIENTE, RESTAURANTE/VENDEDOR, REPARTIDOR, ADMINISTRADOR**) y determina los permisos de acceso a las distintas funcionalidades.

Las funcionalidades indispensables de este servicio son las siguientes:

- **Registro de Usuarios**
  Permite que nuevos usuarios se registren en la plataforma proporcionando su información básica (email, contraseña y rol). Durante este proceso, el servicio se encarga de validar que el correo no esté repetido, encriptar la contraseña y almacenar el usuario en la base de datos.
- **Login (Inicio de Sesión)**
  Permite que un usuario existente ingrese a la plataforma utilizando su email y contraseña. El servicio verifica las credenciales proporcionadas y, si son correctas, continúa con la generación del token de acceso.
- **Generación de JWT**
  Una vez validadas las credenciales, el servicio genera un JSON Web Token (JWT) que contiene la información relevante del usuario (Id, email, rol, etc.). Este token será utilizado por el resto de los microservicios para autenticar y autorizar las peticiones que realiza el usuario.
- **Validación de Credenciales y Tokens**
  El Auth-Service es responsable de validar las credenciales durante el login y de validar los JWT enviados en las peticiones hacia otros servicios. Esto asegura que solo usuarios autenticados y con el rol adecuado puedan acceder a las funcionalidades protegidas de la plataforma.

Restaurant-Catalog-Service:

El servicio de Catálogo de Restaurantes está destinado principalmente para los roles de ADMINISTRADOR y CLIENTE. Su objetivo es gestionar la información de los restaurantes y sus menús, y exponerla de forma adecuada para que los clientes puedan consultarla. Las funcionalidades indispensables de este servicio son las siguientes:

- **Gestión de Restaurantes (CRUD) – Rol: ADMINISTRADOR**: El administrador debe poder crear, leer, actualizar y eliminar restaurantes dentro de la plataforma. Esto incluye datos como nombre del restaurante, dirección, horarios de atención, información de contacto.
- **Gestión de Ítems de Menú (CRUD) – Rol: RESTAURANTE:** El servicio debe permitir crear, leer, actualizar y eliminar los ítems de menú asociados a cada restaurante, incluyendo nombre del platillo, descripción, precio y disponibilidad.
- **Listado de Restaurantes Disponibles – Rol: CLIENTE**: El cliente debe poder consultar el catálogo de restaurantes disponibles visualizando la información necesaria para decidir dónde ordenar (por ejemplo, nombre, tipo de comida, horario, calificación, etc.).
- **Listado de Menú por Restaurante – Rol: CLIENTE**: Una vez el cliente selecciona un restaurante, el servicio debe permitir listar el menú disponible de dicho restaurante, mostrando los ítems activos, sus precios y descripciones.

Order-Service:

El servicio de Ordenes está destinado para los roles de CLIENTE y RESTAURANTE, las funcionalidades indispensables de este servicio son las siguientes:

- **Realizar Orden:** El usuario al momento de tener los productos requeridos en el carrito debe de poder realizar la orden de esos productos y deberá de ser enviada al restaurante.
- **Cancelar Orden:** Si el cliente decide cancelar la Orden por algún motivo este debe de poder actualizar el estado de la orden y actualizar el estado de la orden para el restaurante a CANCELADO.
- **Recibir Orden:** En el momento de que el cliente cree una orden el restaurante debe de poder visualizar las ordenes creadas por los clientes y seleccionar una orden para comenzar a realizarla lo cual marcará el estado de la Orden Cómo EN PROCESO y

una vez finalizada el restaurante deberá de actualizar el estado manualmente a FINALIZADO.
- **Rechazar Orden:** En dado caso el restaurante no tenga suficiente stock de producto o se encuentre con poca disponibilidad de personal el restaurante debe de poder rechazar la orden, lo cual marcará el estado de la orden para el cliente cómo RECHAZADA.

Delivery-Service:

Delivery Service es un servicio orientado para los repartidores, las funcionalidades claves de este servicio son:

- **Aceptar un pedido:** En el momento de que un restaurante marque una Orden como LISTA los repartidores podrán aceptar una de estas ordenes para poder iniciar le entrega lo cual marcará la orden con el estado de EN CAMINO.
- **Actualizar estado de la entrega:** En el momento en el que el repartidor haga entrega del pedido este debe de actualizar el estado del pedido a ENTREGADO, en dado caso ocurra algún percance el repartidor debe de actualizar el estado del pedido cómo CANCELADO.

Notification-Service:

Notificación Service es un servicio orientado únicamente para el rol de **CLIENTE** y para el manejo de notificaciones**,** este servicio notificará al cliente en los siguientes casos:

- **Realizar un pedido:** El usuario debe recibir un correo con un resumen del pedido que acaba de realizar con la siguiente información mínima:
  - Nombre del cliente
  - Numero de Orden
  - Productos ordenados
  - Monto total
  - Fecha de creación de la orden (Timestamp)
  - Estado actual de la orden (CREADA)

- **Cancelar un pedido (Cliente):** El usuario debe de recibir una notificación al momento de cancelar alguna orden y el correo debe de tener la siguiente información mínima:
  - Nombre del cliente
  - Producto o productos ordenados
  - Fecha de cancelación de la orden (Timestamp)
  - Estado actual de la orden (CANCELADA)

- **Envió de la orden:** El usuario será notificado de que la orden ya se encuentre asignada a un repartidor y este se encuentre en camino, la información mínima requerida en el correo es:
  - Numero de orden
  - Nombre del repartidor a cargo
  - Productos ordenados
  - Estado actual de la orden (EN CAMINO)

- **Cancelación del pedido (Restaurante y repartidor):** Si el restaurante o el repartidor desea cancela la orden por alguna razón el cliente debe de ser notificado por correo y la información mínima es la siguiente:
  - Nombre del Restaurante/Vendedor o Nombre del repartidor
  - Razón de la cancelación
  - Productos ordenados
  - Estado actual de la orden (CANCELADA)

- **Rechazar Orden:** En el momento de que un restaurante decida rechazar una orden

este debe de informar al cliente por medio de correo que su orden fue rechazada, el contenido mínimo del correo debe de ser:

- Nombre del restaurante
- Número de orden
- Productos de la orden
- Estado de la orden (RECHAZADA)

4.3 Requerimientos técnicos

Herramientas permitidas	Tipo
React, Angular, Vue.js, JS Vanilla, etc	Frontend
TypeScript, JavaScript con NestJS, Express, Fastify, Hapi	Backend (Node)
Java con Spring Boot, C#/.NET, ASP.NET, Go con GIN/FIBER, python con FastAPI, Flask o Django	Backend (Alternativas)
MySQL, MSSQL, Oracle Database, PostgreSQL, MariaDB	Base de datos
Google Cloude, AWS, Microsoft Azure	Nube (Recomendado GCP)
GitHub	Control de versiones
Docker, Docker Compose	Contenedores

4.4 Entregables

- **Documentación completa**
- **Código fuente de la aplicación**
- **Archivos de configuración utilizados (Json, yml, etc)**
- **Tag con el formato vMajor.Minor.Patch**

- **Fecha límite de entrega:** 19 de febrero a las 23:59.
- **Colaborador obligatorio en el repositorio:** Samashoas
- **Nombre del repositorio:** *SA_PROYECTO_Carnet*.
- **Medio de entrega:** A través de la plataforma UEDI

5. Desarrollo de Habilidades Blandas

5.1 Proyectos Individuales

El proyecto representa una oportunidad para desarrollar autonomía y autoevaluación. Cada estudiante asumirá la totalidad de las fases del proyecto: investigación, desarrollo, pruebas y documentación

5.1.1 Autogestión del Tiempo

El estudiante deberá planificar sus tiempos de trabajo, dividir sus actividades por fases y cumplir con las fechas de entrega.

5.1.2 Responsabilidad y Compromiso

Al trabajar de forma independiente, el estudiante asume completa responsabilidad por la calidad y completitud del proyecto. Esto fomenta el compromiso con el aprendizaje autónomo, la ética profesional y la entrega de productos funcionales.

5.1.3 Resolución de Problemas

El proyecto individual exige que el estudiante resuelva desafíos técnicos por su cuenta, lo que impulsa la creatividad, la investigación y la toma de decisiones lógica ante errores o problemas emergentes durante el desarrollo.

5.1.4 Reflexión Personal

Al finalizar el proyecto, el estudiante deberá realizar una autoevaluación crítica en la que reflexione sobre lo aprendido, los retos enfrentados y las habilidades que necesita mejorar.

6. Cronograma

Tipo	Fecha Inicio	Fecha Fin
Asignación de Proyecto	22/01/2026	19/02/2026
Elaboración	22/01/2026	19/02/2026
Calificación	20/02/2026	21/02/2026

7. Rúbrica de Calificación

7.1 Requisitos para optar a la calificación

Antes de ser evaluado, el proyecto deberá cumplir con los siguientes requisitos mínimos. Si alguno de estos no se cumple, el proyecto no podrá ser calificado y se considerará como no entregado o incompleto:

Tema	Descripción	Cumple (Sí/No)
Gestión y entregas del proyecto	Se deben haber entregado y calificado las primeras 3 practicas del laboratorio para tener derecho a calificación de la fase 1	

7.2 Detalle de la Calificación

Descripción	Valor	Punteo
Frontend	**20**	**-**
Consumo de servicios desde el frontend	10	
UI coherente con toda la aplicación	2.5	
UX amigable con el usuario	2.5	
Respuesta y navegación rápida	5	
Funcionalidades	**40**	**-**
Manejo de JWT para las sesiones	3	
Registro de cliente	1.5	
Registro de Restaurante	1.5	
Registro de Repartidores	1.5	
Creación de orden	1	
Creación de productos nuevos	1.5	
Actualizar productos	1	
Eliminar producto	1	
Actualización de estado de orden (Cliente)	1	
Actualización de estado de orden (Restaurante)	2	
Actualizar estado de orden (Repartidor)	1	
Listado de ordenes por parte del restaurante	1	
Cancelar Orden (Cliente)	1	
Cancelar Orden (Restaurante)	1	
Cancelar Orden (Repartidor)	1	
Notificación de orden creada	4	
Notificación de orden cancelada (Cliente)	4	
Notificación de orden en camino	4	
Notificación de orden cancelada (Restaurante)	4	
Notificación de orden cancelada (Repartidor)	4	
Arquitectura e infraestructura	**30**	**-**
Implementación de microservicios	6	
Creación de imágenes funcionales y optimizadas de Docker	6	

Uso de GCP para el despliegue en la nube de la aplicación	6	
Conexión correcta de los componentes de la aplicación (Base de datos, frontend, backend)	6	
Uso de API GATEWAY para acceso a los microservicios	6	
Uso de gRPC para comunicación entre microservicios	6	
Preguntas	**10**	**-**
Pregunta 1	2	
Pregunta 2	2	
Pregunta 3	2	
Pregunta 4	2	
Pregunta 5	2	
Penalizaciones		
La aplicación no fue desplegada en GCP	-5%	
El auxiliar no fue agregado al repositorio	-10%	
Entrega fuera del plazo establecido	-20%	
No hay persistencia de datos	-40%	
No utilizó una arquitectura orientada a microservicios	-50%	
No se demuestra trabajo progresivo	-50%	
La aplicación no fue desplegada en nube	-60%	
TOTAL	**100**	

7.3 Valores

En el desarrollo del proyecto se espera que los estudiantes demuestren un alto nivel de honestidad académica, responsabilidad ética y compromiso profesional. El cumplimiento de los principios que se detallan a continuación será obligatorio y cualquier incumplimiento será sancionado conforme a las normativas vigentes de la Escuela de Ciencias y Sistemas.

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**
   Queda estrictamente prohibido copiar, replicar o adaptar total o parcialmente el código, documentación, lógica o cualquier componente del proyecto desde otras fuentes sin el debido análisis, modificación y referencia.

   - La detección de plagio (entre compañeros, de internet o de ciclos anteriores) será penalizada con calificación de 0 puntos.
   - Los responsables serán **reportados formalmente a la Escuela de Ciencias y Sistemas**.
   - Esto incluye el uso de ediciones superficiales de código ajeno, sin comprensión real ni justificación lógica.

3. **Uso Responsable de Recursos Externos**
   Está permitido el uso de bibliotecas de consulta, ejemplos o fragmentos educativos siempre y cuando:

   - Se referencien adecuadamente en el código o en los anexos.
   - Se comprenda completamente su funcionamiento.
   - No se utilicen como sustituto de la lógica propia del proyecto. Cualquier duda deberá ser consultada con el catedrático o auxiliar antes de su uso.

4. **Revisión y Validación del Trabajo**
   El equipo docente podrá utilizar herramientas automáticas y revisiones manuales para comparar entregas y detectar similitudes no justificadas.

   - En caso de sospecha, el estudiante deberá defender su solución, explicar sus decisiones y justificar el funcionamiento de su código.
   - Si no logra demostrar la autoría o comprensión del trabajo, se asignará **una calificación de 0 puntos**, sin opción a apelación académica.

5. **Entrega y Fechas Límite**
   - No se permitirá realizar modificaciones al código ni a la documentación después de la fecha de entrega final establecida en el cronograma.
   - Las entregas fuera del plazo establecido pueden considerarse no válidas y se aplicarán las penalizaciones indicadas, a menos que se haya aprobado una prórroga justificada con anterioridad.