SISTEMA DE PEDIDOS EN LÍNEA PARA LAS CAFETERÍAS DE LA UNIVERSIDAD EAN

UNIVERSIDAD EAN

27/05/2025

ESTUDIANTES

Madison Castro Villamil

Juan Felipe Arias Aranguren

Simón Daniel Kostial Pinilla

Néstor Arley Ramírez Ramírez

CURSO

Bases de datos

PROFESOR

John Jairo Porras Vega

PROYECTO INTEGRADOR: SISTEMA DE PEDIDOS EN LÍNEA PARA LAS CAFETERÍAS DE LA UNIVERSIDAD EAN

GENERALIDADES DEL PROYECTO

Nombre del Proyecto

Sistema de Pedidos en Línea para las Cafeterías de la Universidad EAN

Contexto del Escenario

Las cafeterías de la Universidad EAN enfrentan problemas de largas filas y demoras en la atención, lo que afecta la experiencia del usuario y la eficiencia operativa. La implementación de un sistema de pedidos en línea permitirá a los estudiantes realizar sus pedidos de forma anticipada, reducir los tiempos de espera y optimizar la organización del personal de las cafeterías.

Descripción Breve

El sistema consistirá en una plataforma web y móvil que permitirá a los estudiantes de la Universidad EAN seleccionar productos del menú, programar horarios de recogida y efectuar pagos en línea, o al momento de retirar su pedido. La cafetería recibirá la orden y podrá prepararla con anticipación, optimizando los tiempos de entrega y mejorando la gestión de recursos.

Objetivo General

Crear un sistema de pedidos en línea para las cafeterías de la Universidad EAN que ayude a reducir los tiempos de espera, mejorar la organización del personal y brindar una mejor experiencia a los estudiantes a través de una plataforma web y móvil fácil de usar, segura y eficiente.

Objetivos Específicos

- Diseñar una plataforma web y móvil donde los estudiantes puedan ver el menú, hacer pedidos y programar la hora de recogida.
- Implementar un sistema de pago en línea seguro y confiable para facilitar las transacciones.

- Incorporar notificaciones en tiempo real para que los usuarios sepan cuándo su pedido está listo.
- Automatizar la gestión de pedidos y el control de inventario para mejorar la eficiencia en las cafeterías.
- Asegurar que la plataforma sea rápida, segura y fácil de usar, garantizando una buena experiencia para los estudiantes.
- Obtener retroalimentación de los usuarios, para evaluar y efectuar mejoras constantes para optimizar el sistema.

Alcance del Proyecto

Incluye:

- Plataforma web y móvil.
- Gestión de pedidos.
- Sistema de pago en línea.
- Notificaciones en tiempo real de pedidos listos.
- Anuncios y alertas.
- Historial de pedidos.

No incluye:

- Logística de entrega a domicilio.
- Pagos de servicios de pasarela
- Integración con otras plataformas de pedidos externas.

REQUISITOS DEL SISTEMA

Requisitos Funcionales:

- Registro y autenticación de usuarios a través de nombre de usuario y contraseña.
- Visualización del menú actualizado.
- Realización y programación de pedidos en línea.
- Notificaciones en tiempo real sobre el estado del pedido.
- Sistema de pago en línea mediante tarjeta o billeteras digitales.
- Generación de reportes de ventas y análisis de datos.

Requisitos No Funcionales:

- Seguridad en la autenticación y transacciones de pago.

- Disponibilidad de la plataforma de manera continua en los horarios en los que las cafeterías de la Universidad EAN están abiertas.
- Interfaz amigable e intuitiva (usabilidad).
- Escalabilidad para soportar un alto número de usuarios simultáneos.
- Tiempos de respuesta menores a 3 segundos.

PLAN DE IMPLEMENTACIÓN Y ENTREGAS

Plan de Trabajo del Proyecto:

- **Fase 1 (Semana 1-2):** Análisis de requisitos y diseño de la base de datos.
- **Fase 2 (Semana 3-4):** Desarrollo del backend y estructura del sistema.
- **Fase 3 (Semana 5-6):** Desarrollo del frontend y pruebas iniciales.
- **Fase 4 (Semana 7-8):** Integración y pruebas finales.
- **Fase 5 (Semana 9):** Implementación y despliegue.

Entregables Principales:

- Documento de especificaciones del proyecto.
- Diseño de base de datos.
- Prototipo funcional de la plataforma.
- Informe de pruebas.
- Documentación del código y manual de usuario.

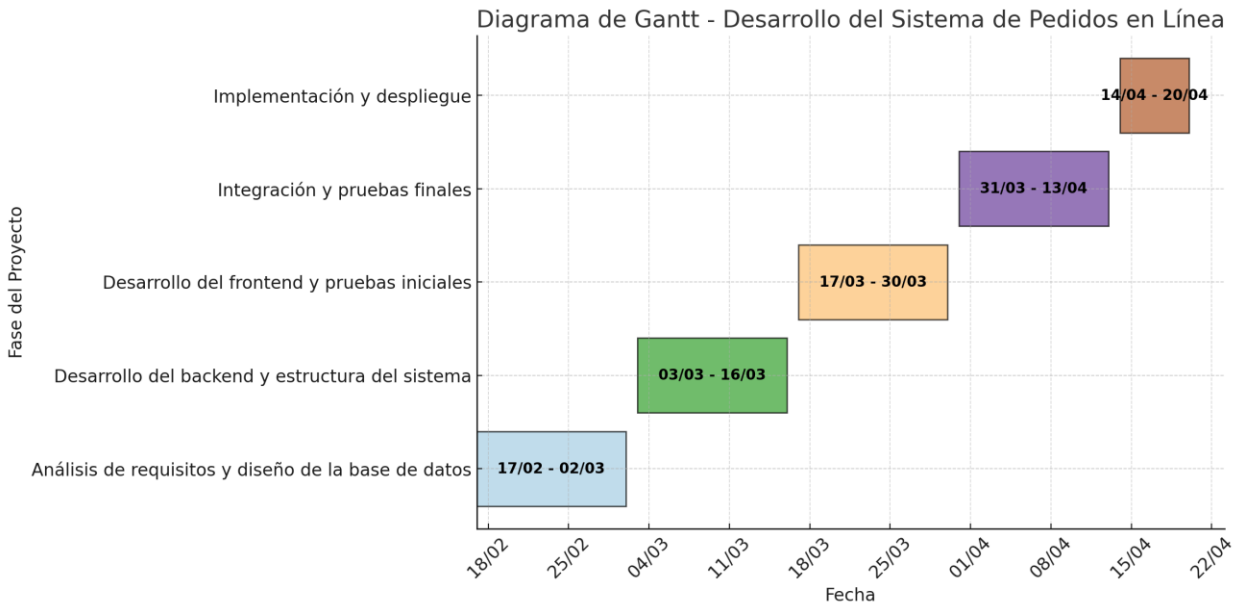
PLAN DE PRUEBAS

Se contemplan los siguientes casos de prueba:

1. **Registro y autenticación:** Verificación del correcto inicio de sesión a partir de credenciales (usuario y contraseña) y recuperación de contraseña.
2. **Realización de pedidos:** Comprobación del flujo de selección de productos y programación de pedidos.
3. **Notificaciones en tiempo real:** Validación de la recepción de alertas cuando el pedido esté listo.
4. **Sistema de pagos:** Pruebas de seguridad y procesamiento correcto de transacciones teniendo en cuenta el método de pago (tarjeta, Nequi, Daviplata, efectivo, etc.).
5. **Carga y rendimiento:** Simulación de múltiples usuarios simultáneos para medir tiempos de respuesta.

Diagrama de Gantt

Fase	Inicio	Fin	Entregables
Análisis de requisitos y diseño de la base de datos	17/02/2025	02/03/2025	Documento de requerimientos y esquema de la base de datos
Desarrollo del backend y estructura del sistema	03/03/2025	16/03/2025	API funcional y conexión con la base de datos
Desarrollo del frontend y pruebas iniciales	17/03/2025	30/03/2025	Prototipo funcional con interfaz gráfica
Integración y pruebas finales	31/03/2025	13/04/2025	Sistema completamente integrado y pruebas de calidad
Implementación y despliegue	14/04/2025	20/04/2025	Implementación final y documentación del proyecto



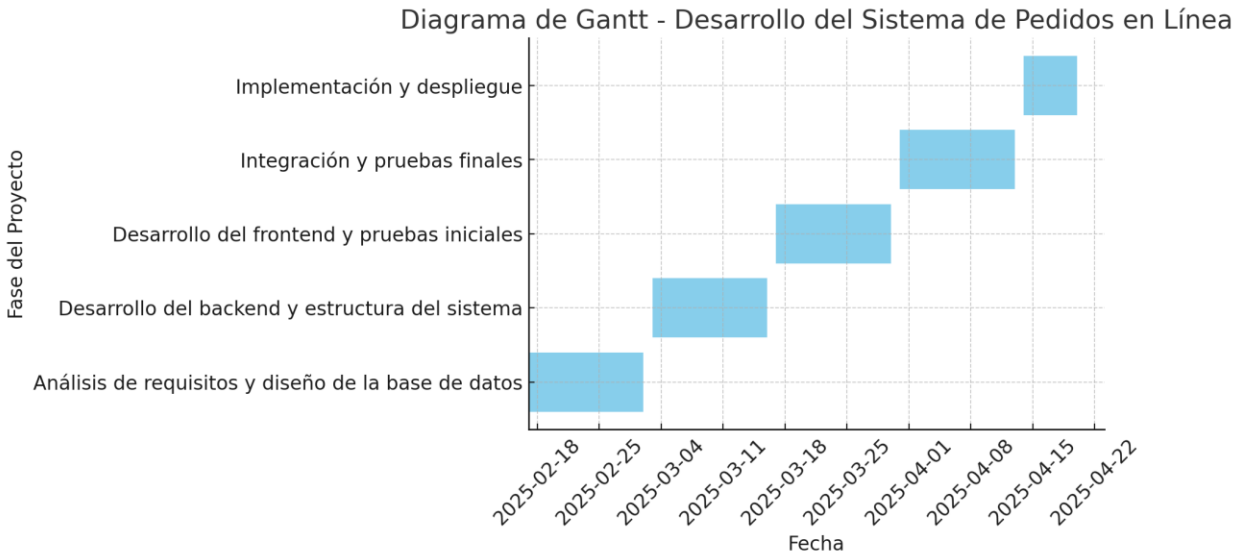


Diagrama de Gantt detallado:
https://universidadeaneduco-my.sharepoint.com/:x:/g/personal/skostia57880_universidadean_edu_co/EbYXu1-k5mBFskpkAHAkMWwBo5kn8dkWxMZ1DN3TWK00Ug?rtime=iuECRtRO3Ug

Requisitos Específicos del Sistema:

1. Gestión de Usuarios:

- **Registro de Usuarios:**

El sistema debe permitir el registro de nuevos usuarios mediante un formulario que capture: nombre, apellido, correo electrónico, número de teléfono y contraseña.
Los correos electrónicos deben ser únicos en el sistema.

- **Autenticación de Usuarios:**

Los usuarios deben poder acceder al sistema utilizando su correo electrónico y contraseña.
Roles de Usuario: Los usuarios deben tener roles como estudiantes, profesores, colaboradores. Esto permitirá la gestión de permisos y restricciones.

2. Gestión de Productos:

- **Menú Actualizado:**

Los productos deben poder ser añadidos al sistema con nombre, descripción, precio, stock y categoría.
El sistema debe permitir al administrador editar y eliminar productos del menú.
La categoría de los productos debe ser gestionable y asignada en el momento de la creación.

3. Realización de Pedidos:

- **Proceso de Pedido:**

Los usuarios deben poder seleccionar productos del menú, ingresar la cantidad deseada y programar la hora de recogida del pedido.
El sistema debe almacenar los detalles del pedido como estado, fecha de creación, total y un número de identificación de pedido.

- **Notificación de Estado de Pedido:**

El sistema debe enviar notificaciones en tiempo real al usuario cuando el pedido haya cambiado de estado, desde pendiente hasta listo para recoger.

4. Sistema de Pagos:

- **Procesamiento de Pagos:**

Los usuarios deben poder pagar sus pedidos mediante tarjetas de crédito, efectivo, o billeteras digitales como Nequi y Daviplata.
El sistema debe actualizar el estado del pedido a pagado una vez que la transacción haya sido completada con éxito.
Se debe proporcionar la opción de pago en efectivo con un estado de pago pendiente, que se actualice al momento de la recogida del pedido.

5. Reportes de Ventas y Análisis de Datos:

- **Generación de Reportes:**

El sistema debe ser capaz de generar reportes de ventas diarias, productos más vendidos y análisis de transacciones.
Los reportes deben estar disponibles tanto para los administradores como para los usuarios con roles específicos.

Requisitos No Funcionales:

- **Rendimiento y Escalabilidad:**

Optimizar las consultas a la base de datos, implementando índices adecuados para las tablas más consultadas (Ej. Productos, Pedidos, Usuarios).

- **Usabilidad:**

La interfaz debe ser clara, intuitiva y funcional, de modo que los usuarios puedan navegar fácilmente entre las opciones de registro, pedidos, pago y reportes. Debe poder utilizarse de un dispositivo móvil.

- **Disponibilidad:**

El sistema debe estar disponible 24/7, garantizando que los usuarios puedan realizar pedidos en cualquier momento durante las horas de funcionamiento de las cafeterías.

- **Tiempos de Respuesta:**

Las consultas y respuestas del sistema deben completarse en un máximo de 3 segundos por cada acción realizada por el usuario (registro, inicio de sesión, pago).

- **Seguridad:**

La aplicación dependiendo de si se inicia como usuario o como empleado debe poseer credenciales de inicio de sesión, donde el usuario se definirá como la cédula de la persona y la contraseña la elegirá el usuario/empleado al registrarse. Adicionalmente, solo los empleados tendrán acceso a las bases de datos de sus respectivas cafeterías y será pertinente implementar un 2fa para estos. Los usuarios solo tendrán acceso a los pedidos que involucren su ID de usuario y a la base de datos de productos de cada cafetería para poder realizar sus pedidos.

Requisitos de la Base de Datos

Sumado a los requisitos funcionales inherentes del software mencionados previamente se establecen los siguientes requisitos solamente considerando las interacciones con la base de datos:

Tablas / Entidades:

La base de datos debe contar de manera inicial con las siguientes 8 tablas, siendo la existencia inicial de estas tablas un requisito funcional de la BD:

Usuarios: ID_usuario (INT, llave primaria), nombres, apellidos, cédula, teléfono, correo, contraseña. (definidos como VARCHAR con ajustes a la longitud del mismo según sea necesario).

Ninguno de estos campos puede quedar como NULL a excepción de dos posibles campos; segundo nombre y segundo apellido, en caso de que se decidan implementar las mismas.

Pedidos: ID_pedido (INT, llave primaria), estado del pedido (VARCHAR variable según se requiera), fecha del pedido (DATE en formato DD-MM-AAAA) y valor total a pagar por el mismo (DECIMAL(12,2) calculado a partir de la suma del precio de cada producto * la cantidad solicitada).

Ningún valor puede ser NULL.

Pagos: ID_pago (INT, llave primaria), fecha del pago (DATE en formato DD-MM-AAAA), método de pago (ENUM con posibles valores Tarjeta (tanto para débito como crédito), Efectivo y Transferencia (este último para Nequi, Daviplata o Coink)) y estado del pedido (ENUM con posibles valores Pagado, Pendiente y Cancelado (anulado)).

Ningún valor puede ser NULL.

Detalles Pedidos: ID_detalles (INT, llave primaria), precio del pedido (DECIMAL (12,2)) y la cantidad de cada producto solicitado (INT). Llaves foráneas ID_producto e ID_pedido.

Ningún valor puede ser NULL.

Productos: ID_producto (INT, llave primaria), nombre del producto (VARCHAR variable según se requiera), stock (INT), descripción (CHAR) y precio unitario (DECIMAL (12,2)). Llave foránea ID_categoría para referenciar a qué categoría pertenece el producto.

Ningún valor puede ser NULL salvo por la descripción.

Categoria: ID_categoria (INT, llave primaria), nombre de categoría (VARCHAR variable según se requiera).

Ningún valor puede ser NULL.

Cafeteria: ID_cafeteria (INT, llave primaria). Nombre, empresa y ubicación (VARCHAR según se requieran).

Ningún valor puede ser NULL.

Empleados: ID_empleado (INT, llave primaria). Nombres, apellidos, cédula, rol, teléfono, correo y contraseña (todos como VARCHAR según se requiera).

Ningún valor puede ser NULL salvo por el posible escenario de segundo nombre y apellido, similar a la tabla Usuarios.

Las relaciones entre estas se definen de manera puntual dentro del modelos entidad-relación y lógico (Data Modeler de Java) adjuntos.

Optimizaciones:

Estas optimizaciones entran dentro de los requisitos no funcionales de la base de datos. No las consideramos como parte esencial de la BD ya que puede funcionar sin estos, sin embargo, pueden ayudarnos a organizar la información de manera más detallada (y posteriormente optimizar consultas).

Indexado: Se realizarán los ajustes de aplicación de índices a medida que se requieran durante los procesos de validación. Inicialmente permanecerán los que se generan automáticamente a través de las llaves primarias.

Apartir de esto se crearon las tablas que conformaran nuestra base de datos en inglés y en mayúsculas como buena práctica de desarrollo.

Si desea conocer el desarrollo completo de la base de datos, puede ingresar al link del repositorio en GitHub: https://github.com/TURINGWORKSCORE/unieats-db.git

EXPLICACION DE CODIGO PRINCIPAL

Creación de la Base de Datos y Tablas

Esta base de datos fue creada en Maria DB utilizando InnoDB como motor de la misma.

Recomendamos hacer lectura de la documentación antes de establecer conexión o ejecutar los scripts sql.

El README.md es un resumen general del proyecto UNIEATS_DB, mostrando las tablas incluidas, requisitos técnicos, y relaciones clave entre tablas de la base de datos relacional. Este se encuentra en la ruta ... dbPj/dbSQL/README.md

El README_TABLAS_UNIEATS.md es la documentación detallada de cada tabla de la base de datos, explicando atributos clave, relaciones y validaciones principales de cada entidad del sistema. Este se encuentra en la ruta ... dbPj/dbSQL/README_TABLAS_UNIEATS.md

- **CREATE DATABASE UNIEATS_DB;**
  - Esta instrucción crea una base de datos llamada UNIEATS_DB, que será el contenedor de todas las tablas y otros objetos.
- **USE UNIEATS_DB;**
  - Selecciona la base de datos UNIEATS_DB para realizar operaciones dentro de ella.

Tabla de Roles (ROLES)

- **CREATE TABLE ROLES**
  - Define la tabla de roles con un ROLE_ID (clave primaria) y ROLE_NAME (nombre del rol), que es único para cada rol (como Estudiante, Profesor, etc.).

Tabla de Usuarios (USERS)

- **CREATE TABLE USERS**
    - Aquí se guardan los datos de los usuarios, con información personal como nombre, dirección y correo electrónico.
    - **ROLE_ID** es clave foránea, relacionada con la tabla ROLES.
    - **Restricciones**:
        - La contraseña debe tener al menos 8 caracteres.
        - Se crea un índice sobre el correo electrónico para búsquedas rápidas.

Tabla de Teléfonos (PHONES)

- **CREATE TABLE PHONES**
    - Guarda los números de teléfono de los usuarios. Relacionada con la tabla USERS a través del campo USER_ID.

Tabla de Cafeterías (CAFES)

- **CREATE TABLE CAFES**
    - Registra las cafeterías, con su nombre, ubicación y nombre de la empresa. Cada cafetería tiene un identificador único.

Tabla de Estaciones de Trabajo (WORKSTATION)

- **CREATE TABLE WORKSTATION**
    - Define las estaciones de trabajo dentro de las cafeterías, como cajero, cocinero, etc.

Tabla de Empleados en Cafetería (EMPLOYEES_CAFE)

- **CREATE TABLE EMPLOYEES_CAFE**
    - Registra a los empleados de cada cafetería con datos personales y roles asignados (relacionados con CAFE_ID y WORKSTATION_ID).
    - Se especifica cuándo un empleado es contratado y cuándo es despedido, así como su estado de actividad.

Tabla de Categorías de Productos (CATEGORIES)

- **CREATE TABLE CATEGORIES**
    - Crea categorías de productos, como bebidas calientes, postres, etc.

Tabla de Productos (PRODUCTS)

- **CREATE TABLE PRODUCTS**
    - Los productos vendidos en las cafeterías se guardan aquí. Cada producto tiene una categoría y una cafetería asignada, con un precio y cantidad en stock.

Tabla de Pedidos (ORDERS)

- **CREATE TABLE ORDERS**
    - Registra los pedidos realizados por los usuarios, especificando el estado del pedido, total de la compra, empleado que tomó el pedido y quien lo entregó.

Tabla de Detalles de Pedidos (ORDER_DETAILS)

- **CREATE TABLE ORDER_DETAILS**
    - Contiene los productos asociados a cada pedido, junto con la cantidad y el precio unitario de cada uno.

Tabla de Métodos de Pago (PAYMENT_METHODS)

- **CREATE TABLE PAYMENT_METHODS**
    - Registra los métodos de pago disponibles (por ejemplo, tarjeta de crédito, efectivo, PSE).

Tabla de Pagos (PAYMENTS)

- **CREATE TABLE PAYMENTS**
    - Aquí se guardan los pagos realizados, asociando cada pago a un pedido, indicando el método de pago, monto pagado y estado del pago.

Inserciones de Datos

- Después de crear las tablas, el código realiza inserciones de ejemplo en las tablas de roles, cafeterías, estaciones de trabajo, usuarios, empleados, productos, pedidos, detalles de pedidos, métodos de pago y pagos.

Ejemplos de inserciones:

1. **INSERT INTO ROLES** agrega roles como "Estudiante", "Profesor", etc.
2. **INSERT INTO USERS** agrega usuarios con nombres, correos y contraseñas, cada uno asignado a un rol específico.
3. **INSERT INTO CAFES** agrega diferentes cafeterías a la base de datos.
4. **INSERT INTO WORKSTATION** agrega las estaciones de trabajo como "Cajero", "Cocinero", etc.
5. Y así sucesivamente para las demás tablas.

Objetivo del Sistema

EL código representa un sistema que maneja:

- Usuarios con diferentes roles.
- Datos de contacto y teléfono de los usuarios.
- Cafeterías con empleados y productos.
- Procesos de pedido y pago, incluyendo detalles de los productos comprados

NOTA IMPORTANTE: En nuestro sistema, decidimos utilizar una base de datos NoSQL para manejar los anuncios y las notificaciones debido a la flexibilidad que ofrece este tipo de tecnología. A diferencia de las bases de datos tradicionales, NoSQL nos permite almacenar datos de forma más dinámica, sin la necesidad de estructurarlos en tablas rígidas. Esto es ideal para un sistema como el nuestro, donde los mensajes pueden variar entre las diferentes cafeterías y pueden cambiar con frecuencia. Además, al usar NoSQL, podemos gestionar de manera más eficiente grandes volúmenes de datos, como las notificaciones que se generan cada vez que un usuario realiza una acción o recibe una actualización en su pedido. Esto nos da la capacidad de escalar fácilmente y de hacer modificaciones rápidas sin tener que rehacer toda la estructura de la base de datos. En resumen, NoSQL nos brinda la flexibilidad y rapidez que necesitamos para ofrecer una experiencia más fluida y personalizada a los usuarios.

EXPLICACION DE CODIGO NOTIFICACION.JSON

README_NOSQL.md

Explica la función de los archivos notifications.json y announcements.json, usados para manejar mensajes y anuncios en tiempo real dentro del sistema NoSQL de UniEats. Este se encuentra en la ruta ... dbPj/dbNoSQL/README_NOSQL.md

1. **CAFE_ID**:
    a. Este es el identificador de la cafetería a la que pertenece la notificación.
    b. Los valores van del 1 al 4, lo que indica que hay 4 cafeterías diferentes con mensajes personalizados.
2. **MESSAGE**:
    a. Es el texto de la notificación que se le envía al usuario o empleado.
    b. Los mensajes pueden ser sobre el estado de un pedido o sobre notificaciones especiales en la cafetería.
3. **TYPE**:
    a. Indica el tipo de notificación.
    b. Los tipos pueden ser:
        i. **estado**: Relacionado con el estado de un pedido (ej., listo para recoger, pago fallido).
        ii. **especial**: Notificaciones especiales sobre eventos o cambios (ej., promociones, cierres, productos agotados, etc.).

Ejemplos de Notificaciones:

- **Tipo "estado"**:
    a. **CAFE_ID 1**: "Tu pedido ya esta listo para recoger."
    b. **CAFE_ID 1**: "El pago no pudo ser procesado. Intentalo nuevamente."
    c. **CAFE_ID 2**: "Tienes un pago pendiente por tu último pedido."
    d. **CAFE_ID 3**: "El producto solicitado está agotado, pero tenemos una alternativa."
- **Tipo "especial"**:
    a. **CAFE_ID 1**: "El buffet de almuerzo está disponible solo hasta las 2 PM."
    b. **CAFE_ID 2**: "Hoy tenemos nueva lasaña con receta especial."
    c. **CAFE_ID 3**: "Promoción: donas 2x1 hasta agotar existencias."
    d. **CAFE_ID 4**: "Cierre anticipado por evento institucional."

FUNCIONAMIENTO

Este archivo JSON podría ser parte de un sistema donde, dependiendo de las acciones que ocurran en las cafeterías (como un pedido listo, pago rechazado, promociones o cambios operativos), el sistema genera notificaciones para los usuarios. Estas notificaciones se categorizan en dos tipos, estado y especial, para diferenciar las situaciones del día a día de las cafeterías (como cuando un producto está agotado) de las comunicaciones especiales (como promociones o cambios de horarios).

EXPLICION DE CODIGO configurationjson.py

1. **Definición de cafeterías:** Se crea un diccionario que asigna a cada cafetería un número (de 1 a 4) y un nombre que representa la ubicación de la cafetería, como "Cafetería Principal", "Cafetería 6 Piso", etc.
2. **Notificaciones generales:** Se define una lista con mensajes generales que se enviarán a los usuarios de todas las cafeterías. Estos mensajes son relacionados con el estado del pedido, como cuando el pedido está listo, si hubo un problema con el pago, o si un producto está agotado.
3. **Notificaciones especiales:** Aquí se define un diccionario en el que cada cafetería tiene su propio conjunto de mensajes especiales. Estos mensajes incluyen promociones, horarios especiales, o cualquier otra información importante específica para cada cafetería.
4. **Creación de las notificaciones:** El código recorre cada cafetería y genera dos tipos de notificaciones:
    a. Las notificaciones generales, que se aplican a todas las cafeterías y son sobre el estado del pedido.
    b. Las notificaciones especiales, que son específicas de cada cafetería y pueden incluir promociones o cambios en el servicio. Estas notificaciones se almacenan en una lista llamada Notifications.
5. **Anuncios promocionales:** De manera similar a las notificaciones, el código también crea una lista de anuncios promocionales para cada cafetería. Estos anuncios pueden incluir

descuentos, productos nuevos, o cambios en el menú de cada cafetería. Los anuncios se almacenan en una lista separada llamada announcement_entries.

FUNCIONAMIENTO

- **Notificaciones**: Estas son enviadas a los usuarios para informarles sobre el estado de su pedido, por ejemplo, si su pedido está listo para recoger, si hay un problema con el pago, o si el producto solicitado está agotado pero se ofrece una alternativa.
- **Notificaciones especiales**: Son mensajes más específicos que están relacionados con eventos especiales, como promociones o cambios temporales en los servicios de la cafetería.
- **Anuncios**: Se envían como promociones o recordatorios especiales, como descuentos, nuevos productos, o cambios en el horario de la cafetería.

EXPLICACION DE CODIGO

Cada objeto representa un anuncio para una cafetería. Cada anuncio tiene tres propiedades principales:

1. **CAFE_ID**: Este campo identifica la cafetería a la que corresponde el anuncio. Los valores de CAFE_ID van de 1 a 4, lo que significa que hay cuatro cafeterías, y cada una tiene sus propios anuncios.
2. **MESSAGE**: Este es el mensaje del anuncio en sí. Los mensajes son promociones, nuevos productos, o información importante relacionada con la cafetería.
3. **TYPE**: El tipo de mensaje. En este caso, todos los mensajes tienen el valor "anuncio", indicando que son promociones o avisos especiales.

Explicación de los datos:

Cafetería 1 (CAFE_ID = 1):

Los anuncios para la **Cafetería Principal** incluyen:

- Promociones de productos como empanadas y bebidas gratis.
- Información sobre el buffet de almuerzo disponible.
- Nuevos productos como arroz chino con pollo.
- Descuentos con tarjeta EAN.
- Avisos sobre el cierre anticipado por jornada de limpieza.

Cafetería 2 (CAFE_ID = 2):

Los anuncios para la **Cafetería 6 Piso** incluyen:

- Combos especiales como lasaña + jugo a un precio reducido.
- Nuevos productos vegetarianos.
- Promociones en pasteles.
- Cambios en el menú, como el cambio del menú de pastas.
- El lanzamiento de un nuevo postre artesanal.

Cafetería 3 (CAFE_ID = 3):

Los anuncios para la **Cafetería 4 Piso** incluyen:

- Ofertas en snacks saludables con descuento.
- Disponibilidad de chocolatinas importadas.
- Anuncios sobre el cierre anticipado por un evento.
- Ofertas como sandwich + té por 5500.
- Galletas de avena recién horneadas.

Cafetería 4 (CAFE_ID = 4):

Los anuncios para la **Cafetería 4-2 Piso** incluyen:

- Promociones en kits de frutas hasta agotar existencias.
- Dulces surtidos a solo 1000 por unidad.
- Anuncio sobre la venta exclusiva de productos empacados.
- Cierre anticipado por mantenimiento de la red eléctrica.
- Nueva presentación de yogur con granola.

¿Cómo funciona este archivo?

Este archivo announcements.json es utilizado para almacenar los anuncios promocionales, y de alertas sobre horarios de cierre y apertura de las cafeterías que serán mostrados a los usuarios. Estos anuncios pueden ser leídos por un sistema (como una aplicación móvil o una página web) y mostrados a los usuarios cuando entren a la plataforma, por ejemplo:

- En una aplicación móvil, podrías tener una sección de "Promociones" donde se muestren estos anuncios según el **CAFE_ID** del usuario.
- En una página web, se pueden cargar estos anuncios dinámicamente dependiendo de la cafetería que se esté visualizando.

TABLAS:

Tabla users

Nombre de campo	Tipo de dato	Llave primaria	Llave foránea	Permite valores nulos	Extra
User_Id	Int	si	No	No	Auto_Increment
First Name	Varchar(50)	No	No	No	
Second_Name	Varchar(50)	No	No	Si	
Last_Name	Varchar(50)	No	No	No	
Second_Last_Name	Varchar(50)	No	No	No	
Cc_Number	Varchar(20)	No	No	No	Unique
Institutional_Email	Varchar(100)	No	No	No	Unique
Address	Varchar(150)	No	No	Si	
Password	Varchar(255	No	No	No	Check(>= 8 caracteres)
Role_Id	Int	No	si	No	FK → Roles.Role_ID
Created_At	Timestamp	No	No	No	DefaultCurrent_Timestamp
Updated_At	Timestamp	No	No	No	On Update Current_Timestamp

Tabla Roles

Nombre de campo	Tipo de dato	Llave primaria	Llave foránea	Permite valores nulos	Extra
Roles_Id	Int	si	No	No	Auto_Increment
Role_Name	Varchar(30)	No	No	No	Unique

Tabla Phones

Nombre de campo	Tipo de dato	Llave primaria	Llave foránea	Permite valores nulos	Extra
Phone_Id	Int	Si	No	No	Auto_Inncrement
User_Id	Int	No	Si	No	Fk->Users.user_Id,On delete cascade

Phone_Number	Varchar(20)	No	No	No	Check(Min caracteres) >= 10

Tabla Cafes

Nombre de campo	Tipo de dato	Llave primaria	Llave foránea	Permite valores nulos	Extra
Cafe_Id	Int	si	No	No	Auto_Increment
Cafe_Name	Varchar(50)	No	No	No	Unique
Cafe_Location	Varchar(100)	No	No	No	
Company_Name	Varchar(100)	No	No	No	Unique

Tabla worstation

Nombre de campo	Tipo de dato	Llave primaria	Llave foránea	Permite valores nulos	Extra
Workstation_Id	Int	Si	No	No	Auto_Increment
Workstation_Name	Varchar(50)	No	No	No	Unique

Tabla Employees_Cafe

Nombre de campo	Tipo de dato	Llave primaria	Llave foránea	Permite valores nulos	Extra
Employees_Id	Int	Si	No	No	Auto_Increment
First_Name	Varchar(50)	No	No	No	
Second_Name	Varchar(50)	No	No	si	
Last_Name	Varchar(50)	No	No	No	
Second_Last_Name	Varchar(50)	No	No	No	
Cc_Number	Varchar(20)	No	No	No	Unique
Email	Varchar(100)	No	No	No	Unique
Address	Varchar(150)	No	No	Si	
Password	Varchar(255)	No	No	No	
Cafe_Id	Int	No	Si	No	Fk->Cafes.Cafe_Id
Workstarion_Id	Int	No	Si	No	Fk->Workstation.workstation_

					Id (On Delete Cascade)
Hired_At	Datetime	No	No	No	Default Current_Timestamp
Fired_At	Datetime	No	No	No	
Is_Active	Boolean	No	No	No	

Tabla Categorías

Nombre de campo	Tipo de dato	Llave primaria	Llave foránea	Permite valores nulos	Extra
Category_Id	Int	Si	No	No	Auto_increment
Category_Name	Varchar(50)	No	No	No	Unique
Cat_Description	Text	No	No	Si	

Tabla Products

Nombre de campo	Tipo de dato	Llave primaria	Llave foránea	Permite valores nulos	Extra
Products_Id	Int	Si	No	No	Auto_Increment
Category_Id	Int	No	Si	No	FK->Category.Category_Id INDEX IDX_CATEGORIES_ID
Cafe_Id	Int	No	Si	No	Fk->Cafe.Cafe_Id INDEX IDX_CAFE_ID
Product_Name	Varchar(100)	No	No	No	
P_Description	Text	No	No	Si	
Price	Decimal(10,2)	No	No	No	Check(Price>0)
Stock Int	Int	No	No	No	Check(stock->=0)

Tabla Orders

Nombre de campo	Tipo de dato	Llave primaria	Llave foránea	Permite valores nulos	Extra

Order_Id	Int	Si	No	No	Auto_Increment
User_Id	Int	No	Si	No	Fk->users.user_Id INDEX IDX_USER_ID
Cafe_Id	Int	No	Si	No	Fk ->Cafes.Cafe_Id INDEX IDX_CAFE_ID
Taken_By_Employee_Id	Int	No	Si	No	Fk->Employees_Cafe.Employees_Id
Order_Date	Datetime	No	No	No	Default Current_Timestamp
Order_status	Enum	No	No	No	Valores='Pending', 'In_Progress','Completed', Cancelled'
Total_Amount	Decimal(10,2)	No	No	No	Check(Total_Amount>0)
Created_At	Timestamp	No	No	No	Default Current_Timestamp
Updated_At	Timestamp	No	No	No	On update Current_Timestamp

Tablas Order_Details

Nombre de campo	Tipo de dato	Llave primaria	Llave foránea	Permite valores nulos	Extra
Detail_Id	Int	Si	No	No	Auto_Increment
Order_Id	Int	No	Si	No	Fk->Orders.order_Id Unique INDEX IDX_ORDER_ID
Product_Id	Int	No	Si	No	Fk->Products.Product_Id Unique

					INDEX IDX_PRODUCT_ID
Quantity	Int	No	No	No	Check (Quantity >0)
Unit_Price	Decimal(10,2)	No	No	No	Check (Unit_Price> 0)

Tabla Payment_Methods

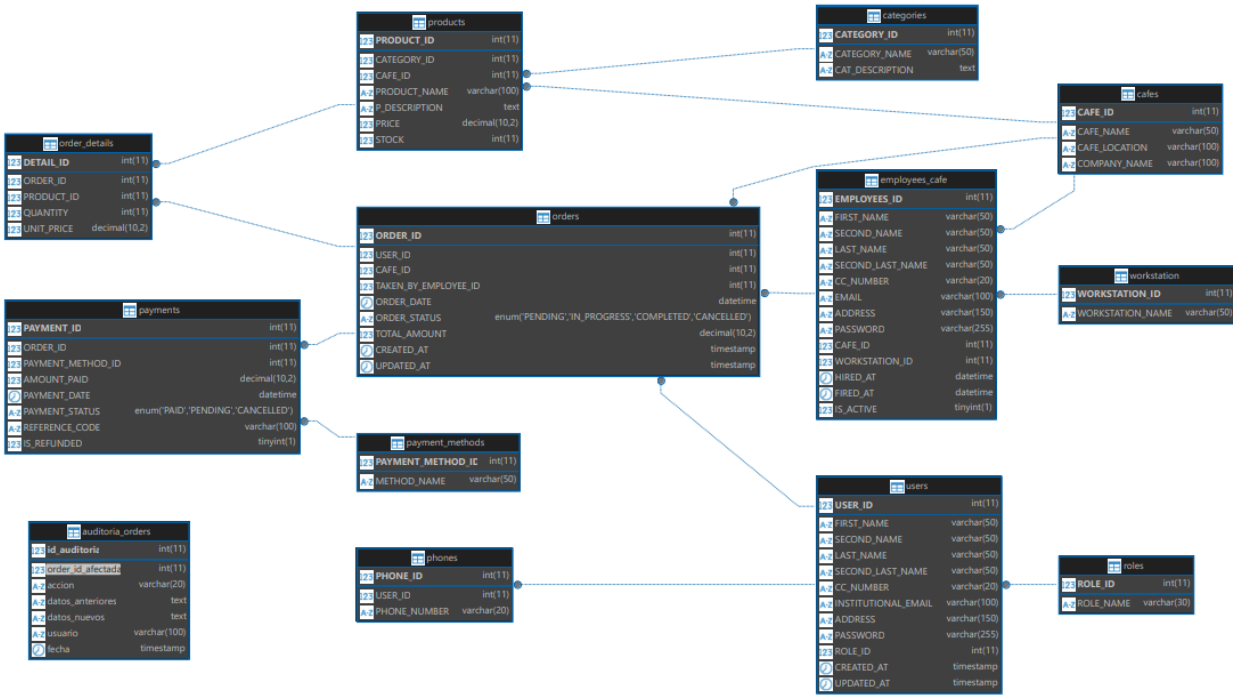
Nombre de campo	Tipo de dato	Llave primaria	Llave foránea	Permite valores nulos	Extra
Payment_Method_Id	Int	Si	No	No	Auto_Increment
Method_name	Varchar(50)	No	No	No	

Tabla Payments:

Nombre de campo	Tipo de dato	Llave primaria	Llave foránea	Permite valores nulos	Extra
Payment_Id	Int	Si	No	No	Auto_Increment
Order_Id	Int	No	Si	No	Fk->Orders.Order_Id
Payment_Method_Id	Int	No	Si	No	Fk->Payment_Methods.Payment_Method_Id On Delete Cascade
Amount_Paid	Decimal(10,2)	No	No	No	Check(Amount_Paid>0)
Payment_Paid	Datetime	No	No	No	Default Current_Timestamp
Payment_Date	Enum	No	No	No	Valores:'Paid','Pending','Cancelled
Payment_Status	Varchar(100)	No	No	Si	Unique
Reference_code	Varchar(100)	no	no	no	unique
Is_Refunded	Boolean	No	No	No	Default False

Tabla auditoria_orders

Nombre de campo	Tipo de dato	Llave primaria	Llave foránea	Permite valores nulos	Extra
Id_Auditoria	Int	Si	No	No	Auto_Increment
Order_Id_Afectada	Int	No	No	Si	
Accion	Varchar(20)	No	No	si	
Datos_Anteriores	Text	No	No	Si	
Datos_Nuevos	Text	No	No	Si	
Usuario	Varchara(100)	No	No	Si	
Fecha	Timestamp	No	No	Si	Default Current_Timestamp



Este diagrama muestra cómo está organizada la base de datos para nuestro sistema de pedidos en línea en las cafeterías de la Universidad EAN. Podemos ver las tablas más importantes, como las de usuarios, roles, pedidos, productos, pagos y empleados, y cómo se conectan entre sí mediante claves que aseguran que la información se mantenga relacionada y consistente. Además la estructura está diseñada para evitar redundancias y facilitar que las consultas a la base de datos sean rápidas y precisas. También incluye una tabla especial para llevar un registro de todos los

cambios que se hagan en los pedidos, lo que nos ayuda a mantener todo claro y controlado durante el funcionamiento del sistema.

Procedimientos:

Hemos utilizado en nuestra base de datos UniEats un total de 8 procedimientos organizados de la siguiente forma.

Poblar_roles(), poblar_workstation(), poblar_categories(), pobla_payment_methods(), que estan encaminados a realizar el llenado o poblamiento inicial de datos y en vista que son 4 diferentes, se ha creado un procedimiento inicial que llama a los primeros 4 para ejecutarlos con una sola orden de llamada, esta función es llamada poblar_parametros().

Para el manejo de inventarios hemos creado dos procedimientos, llamados restar_stock_producto() y aumentar_stock_producto(), los cuales tienen como objetivo restar productos al inventario al momento de las ventas y el segundo es para actualizar los inventarios al momento de recibir una actualización de productos al inventario.

Por último, tenemos crear_orden_con_pago, que es el encargado de realizar el registro de las ventas al momento que el usuario ha realizado su pedido y ha confirmado su compra.

Trigger

Se aplicaron tres triggers para alimentar la tabla **"auditoria_orders"** con el objetivo de registrar toda la actividad relacionada con las órdenes de pedido. Uno registra la información antes de que se cree una nueva orden, otro guarda los cambios realizados cuando se actualizan los pedidos, y el último registra cuando alguna orden es eliminada. De esta forma, mantenemos un control claro y detallado de todas las modificaciones en el sistema.

Los triggers utilizados son: trg_antes_insertar_orders, trg_despues_actualizar_orders y trg_despues_eliminar_orders.

Backend

El backend del proyecto está construido sobre JavaScript con la ayuda de librerías de Node.js (concretamente cors, express, live-server y mysql/mariadb según donde se cree la conexión de la DB).

Cuenta con las conexiones a la base de datos y una api de login para probar las interacciones desde el front hacia la DB pasando por el back.

Frontend

El front del proyecto se conforma por un menú principal en el cual se puede acceder al login de empleado o al login del usuario, cada uno funcionando como un front por aparte.

El front de empleado permite iniciar sesión a un empleado con sus respectivas credenciales (correo y contraseña) y les permite ver los pedidos que se están llevando a cabo, allí el empleado puede marcar como listo el pedido lo cual enviará un correo al estudiante indicandoles que su pedido está listo y finalizar pedido actualizará la tabla con completed.

El front de usuario posee, de igual manera, una pantalla de inicio de sesión para el usuario con sus credenciales (correo y contraseña). Al iniciar sesión, pueden realizar un pedido escogiendo inicialmente la cafetería en la que desea realizar la compra y allí eligen sus productos, la aplicación previsualiza el total del valor de la compra para en la parte final aplicar sobre el botón de hacer pedido donde vuelve a ver nuevamente el valor de la compra acompañado de un resumen de los productos adquiridos, se debe seleccionar el método de pago y se finaliza con el botón de pagar. Habiendo terminado este proceso la aplicación le muestra el número de pedido y ofrece la opción de salir de la app o retornar al inicio para realizar un nuevo pedido.

Nota: Durante el desarrollo de este proyecto integrador, uno de los aprendizajes más significativos que obtuvimos como equipo fue la importancia crucial de la comunicación constante y efectiva entre todos sus miembros. En algunos momentos, enfrentamos dificultades debido a la falta de una coordinación adecuada y la limitada participación de uno de los integrantes, especialmente en periodos de alta carga y tiempos ajustados. Estas situaciones afectaron temporalmente el avance del proyecto y nos enseñaron que mantener un flujo claro de información y compromiso es fundamental para el éxito colectivo.

Afortunadamente, gracias a la intervención del profesor, quien convocó a una reunión para aclarar dudas y realinear esfuerzos, logramos superar estos obstáculos y reenfocar nuestra atención en los objetivos del proyecto. Esta experiencia no solo mejoró la dinámica del equipo, sino que también nos permitió valorar el liderazgo académico y la guía como elementos esenciales en la gestión de proyectos colaborativos. En conjunto, estas lecciones fortalecieron nuestras habilidades para trabajar en equipo, gestionar conflictos y priorizar el cumplimiento de metas en entornos profesionales.