Proyecto Frontend: Plataforma Web para Gestión de Domicilios en Moto

Descripción General del Proyecto

El presente proyecto tiene como objetivo el desarrollo exclusivo del frontend para una plataforma web de gestión de domicilios realizados en motocicleta. La aplicación está dirigida a facilitar la interacción entre restaurantes, clientes, repartidores y operadores logísticos a través de una interfaz moderna, responsiva y accesible desde cualquier dispositivo.

Es importante resaltar que el backend ya se encuentra completamente desarrollado y expone los servicios necesarios a través de una API REST segura. Por tanto, el alcance de este proyecto se limita al diseño e implementación de la interfaz de usuario, así como a la correcta integración con los servicios provistos por el backend.

El código del backend se encuentra en el siguiente repositorio: https://github.com/felipebuitragocarmona/ms_delivery

Historia de usuario

El cliente plantea el desarrollo del frontend de una plataforma de domicilios que ya cuenta con un backend completamente funcional. Esta interfaz debe ser responsiva, intuitiva y moderna, permitiendo la interacción fluida entre restaurantes, clientes, conductores y el sistema de pedidos. El objetivo principal es ofrecer una experiencia eficiente para la gestión y visualización de pedidos de comida entregados por motociclistas.

En esta plataforma, los restaurantes representan los negocios encargados de ofrecer productos alimenticios. Cada restaurante puede tener múltiples productos disponibles, pero también un mismo producto puede ser ofrecido por diferentes restaurantes. Para organizar esta relación muchos a muchos, se ha contemplado una entidad intermedia llamada Menú, que permite asociar productos específicos con restaurantes concretos, facilitando una personalización de la oferta por parte de cada local.

Por su parte, los productos son los elementos que componen la oferta gastronómica. Pueden ser hamburguesas, pizzas, bebidas, entre otros. Estos productos están disponibles en el sistema y se relacionan con los restaurantes a través del menú, permitiendo una administración flexible y centralizada.

Los clientes son los usuarios finales del sistema. Ellos acceden a los menús para seleccionar productos y conformar su pedido. Un mismo cliente puede realizar múltiples pedidos, y un producto del menú puede ser solicitado por muchos clientes. Esta dinámica da origen a la entidad Pedido, que almacena toda la información relacionada con cada orden: qué productos fueron elegidos, en qué cantidades, y a quién se debe entregar.

Cada pedido está vinculado a una única dirección de entrega, lo que establece una relación uno a uno entre ambas entidades. La dirección representa el lugar exacto donde debe entregarse la comida y pertenece exclusivamente a un solo pedido.

Los pedidos deben ser transportados, y para ello, se utilizan motos. Una moto puede transportar muchos pedidos a lo largo del día. Estas motocicletas pueden tener inconvenientes (como accidentes, fallas mecánicas, pinchazos, etc.), y dichos eventos quedan registrados individualmente. Cada inconveniente puede contener fotografías que sirven como evidencia, y estas imágenes están asociadas exclusivamente a un único inconveniente.

Las motos son conducidas por conductores, que pueden rotar entre distintos vehículos. Asimismo, una misma moto puede ser utilizada por varios conductores en diferentes momentos. Esta relación muchos a muchos se organiza a través de una entidad intermedia llamada Turno, la cual registra la fecha y hora de inicio y finalización de cada asignación, permitiendo identificar qué conductor está disponible y qué moto tiene asignada en un momento dado.

Requerimientos Funcionales del Frontend

1. El proyecto debe ser desarrollado en el framework de Angular
2. Interfaz Responsive

   La aplicación debe adaptarse correctamente a dispositivos de escritorio, tablets y teléfonos móviles, garantizando una experiencia de usuario fluida en cualquier resolución.

3. Operaciones CRUD Completas para Todas las Entidades

   Se debe ofrecer al usuario la posibilidad de realizar operaciones de creación, lectura, actualización y eliminación (CRUD) para todas las entidades descritas anteriormente.

   Cada formulario deberá contar con validaciones de campos adecuadas: formatos, obligatoriedad, rangos válidos y consistencia de datos.

4. Autenticación con OAuth

   El sistema debe permitir a los usuarios iniciar sesión a través de:

   - Google
   - Microsoft
   - GitHub

   Al autenticarse, se debe obtener el token de acceso emitido por el proveedor y también la foto de perfil del usuario, la cual deberá ser mostrada en la interfaz.

   El token debe ser enviado en todas las peticiones al backend como parte del encabezado de autorización.

5. El sistema debe de contar con interceptores y guardianes
6. Gráficos e Informes Visuales

   Para propósitos de análisis, la aplicación debe incluir:

   - 3 gráficos circulares
   - 3 gráficos de barras
   - 3 gráficos de series temporales

   Estos gráficos utilizarán datos simulados alojados en un servidor mock (el cual debe de crear los estudiantes).

7. Visualización en Mapa del Estado de Pedidos

   Dentro del módulo de pedidos, se debe incluir un componente de mapa interactivo que permita visualizar en tiempo real la ubicación actual de las motocicletas asignadas a los pedidos.

8. Notificaciones de Nuevos Pedidos

   Cada vez que un nuevo pedido sea asignado a un motociclista, el sistema debe mostrar una notificación visual destacada acompañada de una alerta sonora llamativa que capture de inmediato la atención del usuario.

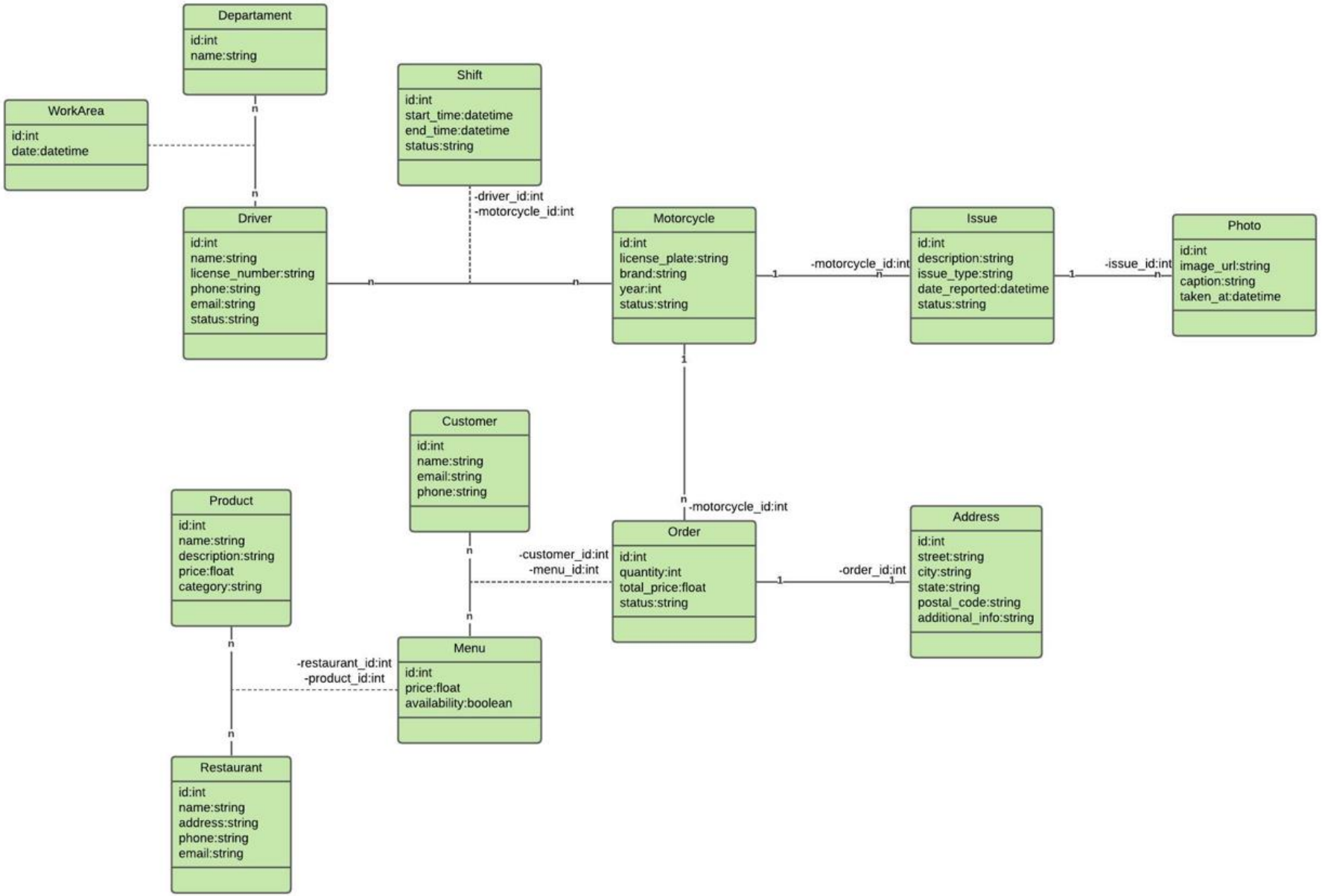
9. Chatbot Inteligente

   El sistema debe contar con un asistente virtual basado en la API de Gemini, que permita responder a preguntas frecuentes del usuario, como:

   - ¿Para qué sirve este sistema?
   - ¿Dónde puedo registrar un nuevo conductor?
   - ¿En qué parte puedo realizar un pedido?

   (Opcional) Como valor agregado se puede incluir:

   Contar con un avatar animado que hable (voz sintética).


Departament
id:int
name:string
WorkArea
id:int
date:datetime
Shift
id:int
start_time:datetime
end_time:datetime
status:string
Driver
id:int
name:string
license_number:string
phone:string
email:string
status:string
-driver_id:int
-motorcycle_id:int
Motorcycle
id:int
license_plate:string
brand:string
year:int
status:string
-motorcycle_id:int
Issue
id:int
description:string
issue_type:string
date_reported:datetime
status:string
-issue_id:int
Photo
id:int
image_url:string
caption:string
taken_at:datetime
Customer
id:int
name:string
email:string
phone:string
Product
id:int
name:string
description:string
price:float
category:string
-motorcycle_id:int
Order
id:int
quantity:int
total_price:float
status:string
-customer_id:int
-menu_id:int
-order_id:int
Address
id:int
street:string
city:string
state:string
postal_code:string
additional_info:string
Menu
id:int
price:float
availability:boolean
-restaurant_id:int
-product_id:int
Restaurant
id:int
name:string
address:string
phone:string
email:string