UNIVERSIDAD NACIONAL DE LOJA

FACULTAD DE LA ENERGÍA LAS INDUSTRIAS Y LOS RECURSOS NO RENOVABLES

CARRERA DE COMPUTACIÓN

PROGRAMACIÓN ORIENTADA A OBJETOS

TEMA: MENU BOARD

INTEGRANTES:

Cristhian Dávila

Ariana Córdova

Miguel Veintimilla

Gerardo Naula

FECHA: 07/02/2025

El presente proyecto de **Gestión de Restaurantes** se lo desarrollo mediante la utilización del **entorno Virtual de Pycharm**. El objetivo principal del proyecto consistió en construir un sistema de gestión para administrar las operaciones de un restaurante, tales como, el control de inventario, reservaciones, gestión de mesas, menús y pedidos.

Inicialmente estos módulos se los dividió entre diferentes grupos de trabajo, de los cuales a nuestro grupo le correspondió el módulo de mesas y reservaciones, el cual se lo trabajo individualmente para después integrarlo o unificarlo con los códigos de los modelos creados por los demás grupos, formando así una sola aplicación.

Además, dentro del proyecto también se consideró la investigación y aplicación de una API, en este caso, sobre Google Maps la cual tiene el propósito de facilitar la ubicación del establecimiento y mejorar la experiencia con usuario.

En esta reflexión, describiremos el proceso de implementación, las decisiones de diseño, los retos enfrentados y las soluciones implementadas.

Implementación de la Interfaz Gráfica

La interfaz gráfica la desarrollamos implementando lo que son los **Templates** junto con **HTML**, **CSS** y **JavaScript** para crear una experiencia de usuario intuitiva y responsive.

La interfaz se estructuró en módulos independientes:

- **Módulo de Reservaciones**: Permite a los clientes seleccionar una fecha, hora y número de personas para reservar una mesa. Se implementó un calendario interactivo para visualizar la disponibilidad de mesas.
- **Módulo de Menús y Pedidos**: Muestra los platos disponibles, organizados por categorías (entradas, platos principales, postres.). Los clientes pueden agregar platos a su pedido y enviarlo.
- **Módulo de Inventario**: Permite a los administradores gestionar el stock de ingredientes y productos. Incluye alertas automáticas cuando el stock está por agotarse.
- **Módulo de Ubicación**: Integra la API de Google Maps para mostrar la ubicación del restaurante y proporcionar indicaciones a los clientes.

La interfaz se conecta con el backend mediante vistas de Django, que procesan las solicitudes y renderizan las plantillas correspondientes.

Uso de la API DE GOOGLE MAPS

La API de Google Maps se integró para mejorar la experiencia del usuario, permitiendo a los clientes localizar el restaurante y obtener

indicaciones precisas. Se utlizó la **Google Maps API** para incrustar un mapa interactivo en la interfaz. Además, se configuró un marcador personalizado para resaltar la ubicación exacta del restaurante.
Para garantizar la seguridad, las claves de API se gestionaron mediante variables de entorno en el archivo settings.py de Django, evitando su exposición en el código fuente.
Mejora la experiencia del usuario al mostrar la ubicación del restaurante de manera clara y precisa lo que permite a los clientes encontrar el restaurante fácilmente.
A continuación, se presenta una guía detallada para obtener tu API Key de Google Maps de forma gratuita:

1. Accede a la Consola de Google Cloud
2. Crea un nuevo proyecto o selecciona uno existente
3. En el menú lateral, selecciona «API y servicios» y luego «Credenciales»
4. Haz clic en «Crear credenciales» y elige «Clave de API»
5. Se te proporcionará tu API Key que podrás copiar y utilizar en tu aplicación

Pruebas de Integración

Durante las pruebas de integración, se utilizó la interfaz de administración de Django para verificar el comportamiento de los modelos y realizar pruebas funcionales en la base de datos. A través del Admin, se identificaron y corrigieron errores en la lógica en el sistema de gestión de restaurantes, asegurando la correcta gestión de datos y relaciones entre entidades.

- **Reservaciones**: Verificar que las reservas se registren correctamente y que se actualice la disponibilidad de mesas.
- **Pedidos**: Comprobar que los pedidos se envíen o realicen correctamente y que el inventario se actualice automáticamente.
- **Google Maps**: Validar que la API de Google Maps muestre la ubicación correcta y que el marcador esté bien configurado.

Estas pruebas permitieron identificar y corregir errores antes de la implementación final, asegurando un sistema robusto y confiable.

Decisiones de Diseño y Cambios Implementados

Durante la planificación del proyecto, se toman diversas decisiones clave para garantizar una implementación eficiente y escalable del sistema de gestión de restaurantes. Algunas de las decisiones más importantes fueron:

1. **Uso de Django como Framework Backend**

   Se eligió Django debido a su estructura basada en el patrón MVC (Modelo-Vista-Controlador), lo que facilitó la organización del código y la gestión de bases de datos mediante su ORM (Object-Relational Mapping). Además, Django proporciona herramientas integradas para autenticación de usuarios y seguridad, lo cual fue un beneficio importante para el proyecto.

2. **Implementación de Django Forms**

Para optimizar la captura y validación de datos en los módulos de reservas y pedidos, se optó por Django Forms en lugar de formularios HTML estándar. Esto permitió manejar la validación de datos de manera más eficiente y reducir errores en la entrada de información.

3. **Uso de plantillas con HTML, CSS y JavaScript**

   Se decidió implementar un sistema basado en plantillas para la interfaz gráfica utilizando el motor de plantillas de Django, junto con CSS y JavaScript. Esto permitió una experiencia de usuario fluida y una interfaz modular, facilitando futuras modificaciones o mejoras sin afectar la funcionalidad general del sistema.

4. **Integración de Google Maps API**

   Se optó por integrar Google Maps API para mejorar la accesibilidad del restaurante para los clientes. Esto permite a los usuarios visualizar la ubicación exacta del establecimiento y obtener direcciones en tiempo real. Para proteger la clave API y evitar accesos no autorizados, se almacenó en variables de entorno dentro del archivo `settings.py`.

Conclusión

Al realizar este proyecto obtuvimos una experiencia entendedora que nos permitió aplicar y consolidar los conocimientos adquiridos a lo largo de la tercera unidad. Aprendimos la importancia de una planificación cuidadosa, la necesidad de pruebas rigurosas y cómo resolver problemas inesperados durante el desarrollo. La integración de múltiples componentes como la interfaz, API externa fue un desafío, pero también una oportunidad para mejorar nuestras habilidades técnicas y de resolución de problemas.