Food Notifier**®**

Informe de Software

**Participantes**

* Pedelhez, Matías Gabriel;
* Soto, Micaela Jazmín Elizabeth;
* X, Gabriela Agustina;
* Zini, Franco Joaquín.

**Descripción del proyecto**

**Food Notifier®** es un producto que se encarga de la administración inteligente de clientes dentro de los comercios. Entre sus funciones permite asignar turnos de forma dinámica a distintos clientes e informar, mediante un dispositivo inalámbrico, el momento en el cual el turno se encuentra disponible.

**Justificación del proyecto**

**Food Notifier®** encuentra su lugar en el mercado permitiendo a las empresas comunicarse con sus clientes de manera efectiva, rápida y de forma enlazada, sin preocuparse por problemas individuales que pudiesen surgir en cada usuario, como ser la imposibilidad de recibir el estado de su pedido (p. ej. si las notificaciones se hiciesen a través de una aplicación móvil y no desde un producto en sí), convirtiéndose en una opción sumamente confiable.

**Planificación del proyecto**

Se realizó una primera reunión mediante la plataforma Discord, con el motivo de debatir ideas sobre la funcionalidad del programa. Una vez decidida fueron planeadas otras reuniones (a través de Meet y Discord), además de la escritura del código y su publicación en la página GitHub para así intervenir en los avances logrados.

**Descripción de etapas 3, 4 y 5**

**Etapa 3: Estructuras compuestas enlazadas (implementación dinámica mediante listas enlazadas)**

Las tareas consistieron en la modificación de la estructura lista implementada de manera estática mediante vectores en la etapa anterior.

Ello implicó la declaración de estructuras de datos de tipo tCola y tNodo. La primera contiene los punteros a los nodos que se encuentran al comienzo y al final de la cola, información importante para efectuar las operaciones de inserción y eliminación de pedidos.

En cuanto a la estructura tNodo, ésta contiene una estructura de tipo tPedidoCliente y un puntero al siguiente nodo de la lista.

Tras la declaración de las variables, se procedió a la adecuación de las funciones de inserción y remoción de nodos. Las mismas son llamadas a través del menú principal. En cuanto a la función de inserción, se denomina agregarElemento y recibe como parámetro un dato de tipo tPedidoCliente, que a su vez fue ingresado por el usuario previamente.

Dentro de dicha función, se procede a la declaración de una variable de tipo tNodo auxiliar para el nuevo nodo. Se reserva memoria mediante la función malloc () y se le asigna valor a los componentes: "datos pedidos" y "siguiente" (en NULL). En cuanto a la actualización de los punteros principio y final de la variable colaPedidos global, ello dependerá de si se está insertando o no el primer elemento. Esa información se obtiene llamando a la función colaVacia ().

Además, en lo que respecta a la operación de remover pedidos, hay tres funciones: eliminarProductoPosicionN (), removerPrimerElemento () y removerElemento(), según se quiera eliminar un pedido en cierta posición de la cola o se encuentre el usuario ante el único pedido o existieren más de uno.

**Etapa 4: Corte de control**

Para esta etapa se pensó en realizar una clasificación de las ventas de acuerdo a intervalos del costo cobrado a los clientes por sus pedidos. Se elaboraron tres categorías de acuerdo al intervalo de precio. Bajo entre $0 y $800, medio entre $800 y $1600 y alto para cobros mayores a $1600. Se calcula cuántas ventas, y promedio recaudado en cada intervalo de precio. Luego se procedió a informar lo total recaudado con todas las ventas y el promedio total de todas las ventas, para tener una idea del gasto en general que hacen los clientes en sus pedidos.

**Etapa 5: Recursividad**

La Recursividad fue implementada en la función menú ()

-**Codificación en lenguaje .c, compilación, ejecución y verificación**

Al igual que en la etapa anterior, se recurrió al método de programación de a pares. En contraste, se incorporó la modalidad de trabajo asincrónica. Ello fue posible gracias al portal GitHub, donde se subió el código en .c del programa, a fin de administrar el proyecto de manera más organizada. Ello permitió que los programadores revisores pudieran ver asincrónicamente el código y efectuar solicitudes de modificaciones. Además, facilitó el trabajo, en tanto tiene la función de comparar versiones, fusionarlas, e incluso restaurarlas.