



Proyecto Final

NetCode

Profesor: Emilio Gabriel Rejón Herrera

Programación Estructurada

LIS – MEFI

Elaboró:

Barón Pat Alan Josue

Cauich Davalos Víctor Enrique

Facultad de Matemáticas

Índice

| Nombre | Página |
|-----------------------------------|---------------|
| Antecedentes de la propuesta | 3 |
| Descripción del Producto Software | 3 |
| Objetivos generales y específicos | 4 |
| Casos de uso | 4 |
| Mapeo de requerimientos | 6 |
| Estándar de Codificación | 9 |
| Modularidad | 13 |
| Proceso de desarrollo | 23 |

Antecedentes de la Propuesta

En México existen muchos puestos que venden distintos tipos de productos en las calles tales como un vendedor ambulante de tamales, hot dogs, chicharrones, helados y entre esos se encuentran los que venden marquesitas, estos productos pese a ser bocadillos simples tienen un potencial comercial aceptable lo que permitiría abrir un establecimiento formal en donde el flujo de clientes sea mayor y por ende las ganancias sean más, pero a todo esto existen momentos en que no se puede controlar correctamente las ventas, lo que ocasionaría un mal manejo de las cuentas y por ende que exista perdidas en las ganancias hasta que cierre el establecimiento, por ello se propone un software sencillo que permita controlar estas situaciones además de ofrecer otros servicios al comprador.

Nombre del Producto Software

Punto de Venta “La Marquesitería”.

Descripción del Producto Software

Es un software enfocado a administrar una marquesitería en donde ofrece un menú en el cual los administradores eligen los productos que los clientes quieren comprar, sean marquesitas, esquites, tostiesquites y con ingredientes a elegir, al igual que se genera un ticket de venta, llevando un control total de las ventas que se produzcan. Por otro lado, será capaz de calcular las ganancias totales que se produzcan en el día, semana o mes. El producto a diferencia de otros puntos de venta es que se centra más que nada en una marquesitería, en la cual se venden esquites, tostiesquites y refrescos, y varían sus pedidos, no como en un supermercado o en una tiendita de la esquina, en la cual son productos ya establecidos y creados, al igual que le ofrece a la empresa una mejor administración en la cual puede acceder un empleado que le ofrece simplemente opciones de venta, y puede acceder un administrador, que a parte que le ofrece lo mismo que a un empleado, le ofrece aún más opciones, como cambiar los precios, personalizar el ticket que se crea, al igual que puede ver sus ganancias por día, por semana y por mes. Se está pensando de igual forma implementar que un empleado solo pueda acceder de una cierta hora hasta otra hora al día, que sería su horario de trabajo, esto para asegurar que, si quiere acceder de nuevo y poner una venta o querer hacer algún cambio sin que un administrador no sepa, el sistema no se lo permitirá, lo cual no lo tiene hasta donde se sabe un punto de venta, ni si quiera de una empresa grande.

Objetivos generales y específicos

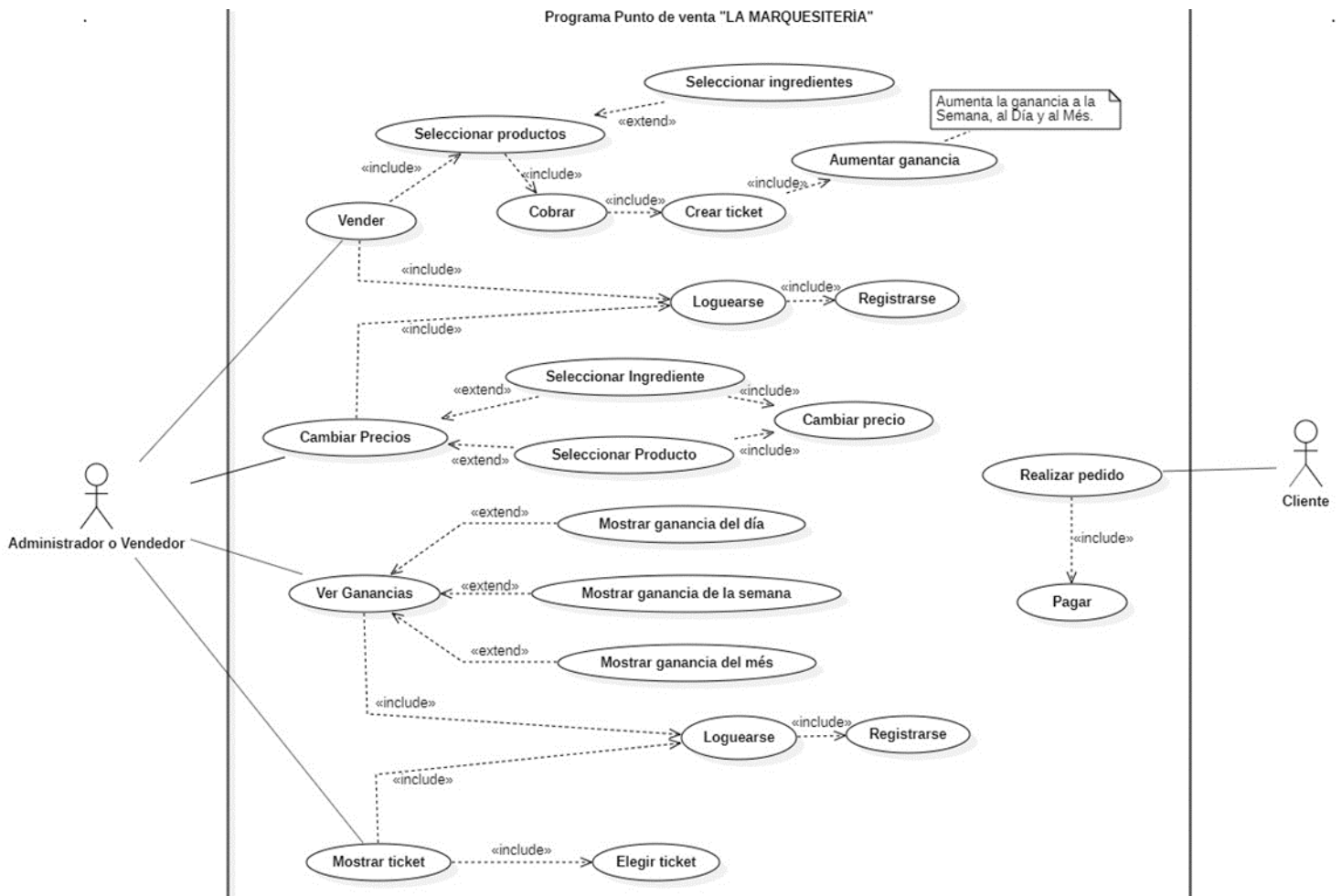
Objetivo general: Incitar a los establecimientos que vendan marquesitas a utilizar este producto para llevar un control sobre el negocio, además de tener formas para calcular las ganancias basándose en los demás servicios proporcionados por el producto.

Objetivos específicos:

- Contiene un menú de usuario y administrador.
- Contiene un menú para agregar ingredientes.
- Elabora tickets de venta.
- Elabora un reporte de ganancias por día.
- Elabora un reporte de ganancias por semana.
- Elabora un reporte de ganancias por mes.
- Cambiar precios.

Casos de Uso

Diagrama General de Casos de Uso



Tipos de Usuario

- **Administrador**: Es el que lleva el control sobre los tipos de productos que se están vendiendo, sobre los ingredientes que se le pueden poner a los productos, la información que se imprime en los tickets y el que administra el control de las ventas y las ganancias que se produzcan, además de revisar que todo opere correctamente respecto al software, es el único que puede cambiar los precios, y esto se podrá saber al momento de loguearse (iniciar sesión), si el usuario y contraseña corresponde al de un administrador, entonces se le dan opciones adicionales a comparación de un empleado normal.
- **Empleado**: El empleado es el que vende y atiende al cliente, lo cual puede entrar al sistema si tiene los datos necesarios, es decir, su contraseña y usuario, al entrar él puede ingresar la cantidad de productos que va a querer al igual que con la cantidad de ingredientes, también con su respectivo ticket se le crea al finalizar el pedido.
- **Cliente**: Es aquel que sólo compra uno o más productos que se ofrezcan en el establecimiento y le añade o no, algún ingrediente del menú, también es aquel que recibe un ticket impreso por el programa con la información respectiva de la orden y el monto a pagar.

Mapeo de Requerimientos

Requerimientos Funcionales

RF_01: Cada vez que un administrador o empleado quiera ingresar, debe validarse por codificación el usuario y contraseña de los mismos, en caso de no existir una cuenta se deberán crear a través del administrador.

RF_02: Al finalizar un pedido, siempre se deberá de realizar un ticket, en el que estén señalados los productos solicitados, así como la cantidad de los mismos, también se deberá señalar el costo individual de cada producto y el costo total que es el que se deberá pagar. Al igual que se debe de guardar cada ticket en un archivo txt.

RF_03: En cualquier momento un administrador puede cambiar precios tanto a productos y a ingredientes que se estén vendiendo en el establecimiento al igual de que cada precio de los productos estará guardado en un archivo diferente de txt.

RF_04: Se podrán ver los tickets que ya se hayan creado, esto eligiéndolo en un interfaz intuitivo, en el cual se pueden consultar los tickets por día, semana o mes respecto al archivo de txt correspondiente siempre y cuando se haya concluido el periodo laboral según el tipo de ticket a consultar.

RF_05: Las ganancias diarias, semanales y mensuales se irán sumando al momento de que se genere cada ticket después de una venta, cada ganancia se irá guardando en archivos txt's respecto al tipo de ganancia que se esté generando al momento, para diferenciarlos se guardaran por día y una vez concluida la semana se guardarán todos los tickets generados por día para que se elabore el de las ganancias semanales, siguiendo el mismo procedimiento para las ganancias mensuales.

RF_06: Las cuentas de los empleados solo podrán acceder al sistema en su horario de trabajo que ha sido prestablecido por el administrador. Esto quiere decir que es un tipo de seguridad para que no puedan acceder en el sistema en cualquier momento.

Requerimientos No Funcionales

RNF_01: El sistema deberá responder de la forma más rápida posible al momento de que se solicite una acción a través de sus interfaces para evitar posibles retrasos en la atención hacia los clientes.

RNF_02: El usuario debe validar su contraseña y el nombre con el que se registró, si no se está registrado, que el administrador debe registrarlo, este no podrá acceder al sistema a realizar actividades. Al igual que cada empleado tendrá un horario en específico de trabajo, esto almacenado por el sistema, en la cual no podrán ingresar en cualquier horario de trabajo.

RNF_03: El sistema debe tener un interfaz en el cual el trabajador o administrador deba poder saber qué hace cada apartado y el cuál se le facilite el poder usar el sistema evitando cualquier posible malinterpretación del mismo que conlleve a un mal uso del software.

RNF_04: El sistema está disponible en cualquier momento, aunque la empresa debe tener la conciencia de solo usarlo en el tiempo laboral a menos que se registre un nuevo trabajador o realizar un cambio de precio.

Casos de uso para cada funcionalidad del sistema

1.- Login / Registro: El usuario proporciona sus datos de sesión como el usuario y contraseña y el programa determinará si los datos de la cuenta están asociados a una cuenta del tipo administrador o el de un empleado.

1.1.- En caso de que alguno de los datos proporcionados este erróneo, el sistema desplegará una alerta de que los datos son incorrectos y se pedirá que los reingrese nuevamente.

1.2.- En el caso de que no exista una cuenta en el sistema le pedirá al usuario que cree una nueva cuenta, que al ser la primera cuenta del sistema está la tomará como una cuenta de tipo administrador.

2.- Cambio de Precios: Una vez confirmado que el administrador haya iniciado la sesión, se le permitirá cambiar los precios de los productos e ingredientes.

2.1.- En el caso de que el administrador por error cambie los precios de algún producto o ingrediente, durante cada cambio aparece una ventana de confirmación.

2.2.- Si el administrador decide no realizar ningún cambio simplemente presiona en el botón de regreso.

3.- Venta de Productos: Las cuentas de administradores y empleados tienen la capacidad de realizar las respectivas transacciones que sus clientes requieran y los tickets generados se almacenan para el cálculo de las ganancias.

3.1.- En el caso de que el cliente no decida agregar o quitar algún ingrediente se pasará inmediatamente a la generación de su ticket de venta y los datos se almacenan.

3.2.- Dado el evento de que no existan suficientes ingredientes o el inventario este vacío se emitirá un cuadro de alerta informando la insuficiencia del mismo.

4.- Visualización de las Ganancias: Cuando el usuario administrador inicie sesión este podrá consultar las ganancias que se general al día, semana o por mes.

4.1.- En el caso de las ganancias por día sólo se podrá consultar después de que finalice el horario laboral.

4.2.- Respecto a las ganancias por día sólo se podrá consultar una vez finalizada la semana o el de la semana anterior.

4.3.- Las ganancias mensuales se visualizarán una vez concluido el mes ó puede consultar la del mes pasado.

5.- Visualización de los tickets: Las cuentas de administradores pueden consultar los tickets generados por el sistema para la corroboración de las ganancias que se generan.

5.1.- En el caso de la consulta de los tickets por día, semana o mes debe haberse concluido dicho periodo laboral, de caso contrario se consultarán tickets previos.

Matriz de requerimientos

| Matriz de requerimientos | | | |
|--------------------------------------|----------------------------|------------------|------------------|
| Requerimientos Funcionales | | | |
| Código | Descripción breve | Prioridad | Objetivos |
| RF_01 | Validación del usuario | Alta | 1 |
| RF_02 | Creación de los tickets | Media | 3 |
| RF_03 | Cambio de precios | Alta | 7 |
| RF_04 | Consulta de tickets | Baja | 3 |
| RF_05 | Ganancias registradas | Alta | 3,4,5 y 6 |
| RF_06 | Horario laboral | Media | 1 |
| Requerimientos No Funcionales | | | |
| RNF_01 | Sensibilidad del sistema | Alta | 3 |
| RNF_02 | Seguridad | Alta | 1 y 7 |
| RNF_03 | Interfaz intuitiva | Media | 1,2,3 y 7 |
| RNF_04 | Accesibilidad del software | Media | 1 |

Definición del estándar de codificación

- El nombre de los identificadores utilizados se nombró de forma representativa para evitar errores de confusión y que indiquen que acciones están realizando, algunos de estos son:
 - Usuario
 - Contraseña
 - Val
 - Val1
 - Token
 - Login
 - Archivo
 - Ultimo
 - Aux
 - Seleccion
 - CantMarquesitas
 - CantIngredientes
 - CantEsquites
 - CantTostiesquites
 - CantToppings
 - CantRefrescos
 - CantAguas
 - Etc.
- Las bibliotecas utilizadas en el proyecto fueron:






























```
1  #include <stdio.h>
2  #include <locale.h> //Para las acentuaciones
3  #include <stdlib.h> //Conversión de tipos de datos
4  #include <string.h> // permite strcmp y strcpy.
5  #include <time.h> //Para obtener fecha y hora de La computadora
6
```

- No se está realizando el uso de macros debido a que los datos se están guardando en archivos.

- Se están utilizando funciones por paso de parámetro o referencia, en donde se llevan a cabo los procesos del sistema como sería en la venta de productos. Algunos de estos, son:

```
14
15 void ValidarUsuario();
16 void MenuAdministrador(char Usuario[15], char Contraseña[15]);
17 void MenuTrabajador(char Usuario[15], char Contraseña[15]);
18 void CrearVenta(char Usuario[15], char Contraseña[15]);
19 int IngresarMarquesitas();
20 int IngresarIngredientes();
21 int IngresarEsquites();
22 int IngresarTostiesquites();
23 int IngresarToppings();
24 int IngresarRefrescos();
25 int IngresarAguas();
26 double CalcularPrecioMarquesitas(int Cant);
27 double CalcularPrecioIngredientes(int Cant);
28 double CalcularPrecioEsquites(int Cant);
29 double CalcularPrecioTostiesquites(int Cant);
30 double CalcularPrecioToppings(int Cant);
31 double CalcularPrecioRefrescos(int Cant);
32 double CalcularPrecioAguas(int Cant);
33 void VerTicket();
34 void VerGanancias();
35 void GananciasDia();
36 void GananciasSemana();
37 void GananciasMes();
38 int CalcularFecha(int *D, int *M, int *A);
39 void CambiarPrecios();
40 void CambiarPrecioMarquesitas();
41 void CambiarPrecioIngredientes();
42 void CambiarPrecioEsquites();
43 void CambiarPrecioTostiEsquites();
44 void CambiarPrecioToppings();
45 void CambiarPrecioRefrescos();
46 void CambiarPrecioAguas();
47 double ValidarPrecio();
48
```

- Los archivos son de uso esencial en el sistema pues se usan para guardar precios, los tickets, ganancias diarias, semanales y mensuales, al igual que los usuarios y contraseñas. Inclusive se dividen por carpetas, como verán a continuación:

| | | |
|---|------------------------|---------------------|
|  Datos | 27/04/2020 11:51 a. m. | Carpeta de archivos |
|  Diagrama | 26/04/2020 08:06 p. m. | Carpeta de archivos |
|  Documentos | 27/04/2020 11:52 a. m. | Carpeta de archivos |
|  Precios | 27/04/2020 11:51 a. m. | Carpeta de archivos |
|  Tickets | 27/04/2020 07:33 p. m. | Carpeta de archivos |
|  LOGIN.txt | 27/04/2020 07:47 p. m. | Documento de te... |
|  PrecioBaseEsquite.txt | 27/04/2020 07:32 p. m. | Documento de te... |
|  PrecioBaseMarquesita.txt | 27/04/2020 07:31 p. m. | Documento de te... |
|  PrecioBaseTostiEsquite.txt | 27/04/2020 07:32 p. m. | Documento de te... |
|  PrecioIngredientesMarquesitas.txt | 27/04/2020 07:32 p. m. | Documento de te... |
|  PrecioRefrescos.txt | 27/04/2020 07:32 p. m. | Documento de te... |
|  PreciosAguas.txt | 27/04/2020 07:32 p. m. | Documento de te... |
|  PrecioToppings.txt | 27/04/2020 07:32 p. m. | Documento de te... |
|  1.txt | 27/04/2020 01:18 p. m. | Documento de te... |
|  2.txt | 27/04/2020 01:19 p. m. | Documento de te... |
|  3.txt | 27/04/2020 01:19 p. m. | Documento de te... |
|  4.txt | 27/04/2020 01:19 p. m. | Documento de te... |
|  5.txt | 27/04/2020 01:19 p. m. | Documento de te... |
|  6.txt | 27/04/2020 01:20 p. m. | Documento de te... |
|  7.txt | 27/04/2020 01:20 p. m. | Documento de te... |
|  8.txt | 27/04/2020 01:20 p. m. | Documento de te... |
|  9.txt | 27/04/2020 01:20 p. m. | Documento de te... |
|  10.txt | 27/04/2020 03:50 p. m. | Documento de te... |
|  11.txt | 27/04/2020 03:50 p. m. | Documento de te... |
|  12.txt | 27/04/2020 03:49 p. m. | Documento de te... |
|  13.txt | 27/04/2020 07:25 p. m. | Documento de te... |
|  14.txt | 27/04/2020 07:29 p. m. | Documento de te... |
|  15.txt | 27/04/2020 07:33 p. m. | Documento de te... |
|  UltimoNoTicket.txt | 27/04/2020 07:33 p. m. | Documento de te... |

- Los comentarios se hacen dentro del código con el uso de la doble barra (//) indicando que función realiza cada parte del código. Por ejemplos, antes de empezar un código, se escribe antes para qué va a servir, como se muestra

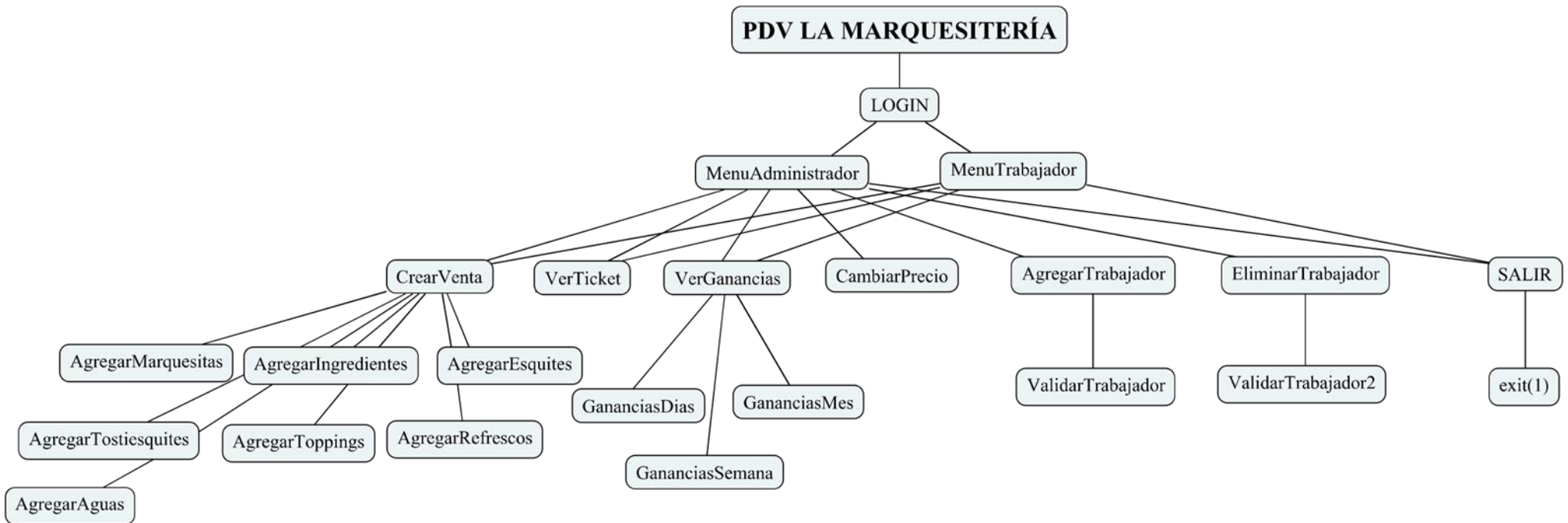
```

47  double ValidarPrecio();
48
49  int main (){
62  //-----Validando usuario y contraseña-----
63  void ValidarUsuario()
64  {
155 //-----Menú del Trabajador-----
156 void MenuTrabajador(char Usuario[15], char Contraseña[15])
157 {
202 //-----Menú del Administrador-----
203 void MenuAdministrador(char Usuario[15], char Contraseña[15])
204 {
259 //-----Función para crear una venta-----
260 void CrearVenta(char Usuario[15], char Contraseña[15])
261 {
557 //-----Vallidación de Marquesitas-----
558 int IngresarMarquesitas()
559 {
571 //-----Vallidación de Ingredientes-----
572 int IngresarIngredientes()
573 {
585 //-----Vallidación de Esquites-----
586 int IngresarEsquites()
587 {
599 //-----Vallidación de Tostiesquites-----
600 int IngresarTostiesquites()
601 {
613 //-----Vallidación de Toppings-----
614 int IngresarToppings()
615 {
627 //-----Vallidación de Refrescos-----
628 int IngresarRefrescos()
629 {
641 //-----Vallidación de Aguas-----
642 int IngresarAguas()
643 {
655 //-----Calcular Precio de Marquesitas-----
656 double CalcularPrecioMarquesitas(int Cant)
657 {
677 //-----Calcular Precio de Los Ingredientes-----
678 double CalcularPrecioIngredientes(int Cant)
679 {
699 //-----Calcular Precio de Esquites-----
700 double CalcularPrecioEsquites(int Cant)
701 {

```

Modularidad

Organización del sistema en general



Organización del código con base en entradas, procesamiento y salidas

Entradas:

1.- Login:

```
62 //-----Validando usuario y contraseña-----
63 void ValidarUsuario()
64 {
65     system("cls");
66     FILE *LOGIN;
67     char *token;
68     char Usuario[15];
69     char Contraseña[15];
70     bool Val = false;
71     int Val1;
72     char Aux[32];
73     int S;
74     printf("%15s", "LOGIN");
75     printf("\n\n%5s%s", "", "USUARIO: ");
76     gets(Usuario);
77     fflush(stdin);
78     printf("\n%2s%s", "", "CONTRASEÑA: ");
79     gets(Contraseña);
80     fflush(stdin);
81     LOGIN = fopen("Datos/LOGIN.txt", "r");
82     if(LOGIN == NULL)
83     {
84         system("cls");
85         printf("¡¡ERROR!!");
86         printf("\nEL ARCHIVO NO EXISTE...");
87         printf("\n\nTeclea ENTER para salir por favor...");
88         fflush(stdin);
89         getchar();
90         exit(-1);
91     }
92     rewind(LOGIN);
93     fgets(Aux, 32, LOGIN);
94     while(!feof(LOGIN) && Val == false)
95     {
96         fgets(Aux, 32, LOGIN);
97         token = strtok(Aux, ",");
98         if(token != NULL){
99             if(strcmp(Usuario, token) == 0)
100             {
101                 token = strtok(NULL, ",");
102                 if(strcmp(Contraseña, token) == 0)
103                 {
104                     Val = true;
105                     token = strtok(NULL, ",");
```

2.- Menús

Administrador:

```
202 //-----Menú del Administrador-----
203 void MenuAdministrador(char Usuario[15], char Contraseña[15])
204 {
205     int Seleccion;
206     do
207     {
208         system("cls");
209         printf("%15s%-15s", "BIENVENIDO: ", Usuario);
210         printf("\n%15s%-15s", "MENÚ DE AD", "MINISTRADOR");
211         printf("\n1. Nueva Venta.");
212         printf("\n2. Ver Ticket.");
213         printf("\n3. Ver Ganancias.");
214         printf("\n4. Cambiar Precios.");
215         printf("\n5. Agregar Trabajador.");
216         printf("\n6. Eliminar Trabajador.");
217         printf("\n7. SALIR.");
218         printf("\nIngrese el número de la opción \nque desee realizar, por favor: ");
219         scanf("%d", &Seleccion);
220         fflush(stdin);
221         switch(Seleccion)
222         {
223             case 1:
224                 CrearVenta(Usuario, Contraseña);
225                 break;
226             case 2:
227                 VerTicket();
228                 break;
229             case 3:
230                 VerGanancias();
231                 break;
232             case 4:
233                 CambiarPrecios();
234                 break;
235             case 5:
236                 break;
237             case 6:
238                 break;
239             case 7:
240                 system("cls");
241                 printf("Que tenga buen día...");
242                 printf("\n\nTeclea ENTER para salir por favor...");
243                 fflush(stdin);
244                 getchar();
245                 exit(1);
```

Empleado:

```
155 //-----Menú del Trabajador-----
156 void MenuTrabajador(char Usuario[15], char Contraseña[15])
157 {
158     int Seleccion;
159     do
160     {
161         system("cls");
162         printf("%15s%-15s", "BIENVENIDO: ", Usuario);
163         printf("\n%15s%-15s", "MENÚ DE T", "RABAJADOR");
164         printf("\n1. Nueva Venta.");
165         printf("\n2. Ver Ticket.");
166         printf("\n3. Ver Ganancias.");
167         printf("\n4. SALIR.");
168         printf("\nIngrese el número de la opción \nque desee realizar, por favor: ");
169         scanf("%d", &Seleccion);
170         fflush(stdin);
171         switch(Seleccion)
172         {
173             case 1:
174                 CrearVenta(Usuario, Contraseña);
175                 break;
176             case 2:
177                 VerTicket();
178                 break;
179             case 3:
180                 VerGanancias();
181                 break;
182             case 4:
183                 system("cls");
184                 printf("Que tenga buen día...");
185                 printf("\n\nTeclea ENTER para salir por favor...");
186                 fflush(stdin);
187                 getchar();
188                 exit(1);
189                 break;
190             default:
191                 system("cls");
192                 printf("ERROR, SELECCIÓN INVALIDA...");
193                 printf("\nIntente de nuevo, por favor.");
194                 printf("\nTeclea ENTER para continuar, por favor...");
195                 fflush(stdin);
196                 getchar();
197                 MenuTrabajador(Usuario, Contraseña);
198                 break;
199         }
200     }
201 }
```


Procesos:

1.- Ventas:

```
259 //-----Función para crear una venta-----
260 void CrearVenta(char Usuario[15], char Contraseña[15])
261 {
262     int CantMarquesitas = 0, CantIngredientes = 0, CantEsquites = 0, CantToppings = 0, CantTostiesquites = 0;
263     int CantRefrescos = 0, CantAguas = 0, Seleccion = 0;
264     bool Val = false;
265     do
266     {
267         fflush(stdin);
268         system("cls");
269         printf("%15s", "SELECCIÓN DE PRODUCTOS");
270         printf("\n1. Ingresar Cantidad de Marquesitas.");
271         printf("\n2. Ingresar Cantidad de Ingredientes de Marquesitas.");
272         printf("\n3. Ingresar Cantidad de Esquites.");
273         printf("\n4. Ingresar Cantidad de Tostiesquites.");
274         printf("\n5. Ingresar Cantidad de Toppings.");
275         printf("\n6. Ingresar Cantidad de Refrescos.");
276         printf("\n7. Ingresar Cantidad de Aguas.");
277         printf("\n8. LISTO.");
278         printf("\n9. CANCELAR.");
279         printf("\nIngrese el número de la opción que desee realizar, \npor favor: ");
280         scanf("%d", &Seleccion);
281         fflush(stdin);
282         system("cls");
283         switch(Seleccion)
284         {
285             case 1:
286                 CantMarquesitas = IngresarMarquesitas();
287                 if(CantMarquesitas == 0)
288                 {
289                     CantIngredientes = 0;
290                 }
291                 break;
292             case 2:
293                 if(CantMarquesitas < 1)
294                 {
295                     printf("ERROR, DEBE AGREGAR AL MENOS 1 MARQUESITA...");
296                     printf("\nTeclea ENTER para continuar, por favor...");
297                     fflush(stdin);
298                     getchar();
299                 }else
300                 {
301                     CantIngredientes = IngresarIngredientes();
302                 }

```

2.- Validaciones de los productos:

```
557 //-----Vallidación de Marquesitas-----
558 int IngresarMarquesitas()
559 {
560     int Num;
561     printf("Ingrese cuántas Marquesitas son: ");
562     scanf("%d", &Num);
563     fflush(stdin);
564     if(Num < 0)
565     {
566         system("cls");
567         Num = IngresarMarquesitas();
568     }
569     return Num;
570 }
571 //-----Vallidación de Ingredientes-----
572 int IngresarIngredientes()
573 {
574     int Num;
575     printf("Ingrese cuántos Ingredientes para Marquesitas son: ");
576     scanf("%d", &Num);
577     fflush(stdin);
578     if(Num < 0)
579     {
580         system("cls");
581         Num = IngresarIngredientes();
582     }
583     return Num;
584 }
585 //-----Vallidación de Esquites-----
586 int IngresarEsquites()
587 {
588     int Num;
589     printf("Ingrese cuántos Esquites son: ");
590     scanf("%d", &Num);
591     fflush(stdin);
592     if(Num < 0)
593     {
594         system("cls");
595         Num = IngresarEsquites();
596     }
597     return Num;
598 }
599 //-----Vallidación de Tostiesquites-----
600 int IngresarTostiesquites()
```

Salidas:

1.- Creaciones de los tickets:

```
655 //-----Calcular Precio de Marquesitas-----
656 double CalcularPrecioMarquesitas(int Cant)
657 {
658     FILE *Costo;
659     char Aux[15];
660     double CostoP;
661     Costo = fopen("Precios/PrecioBaseMarquesita.txt", "r");
662     if(Costo == NULL)
663     {
664         system("cls");
665         printf("¡¡ERROR!!");
666         printf("\nEL ARCHIVO NO EXISTE...");
667         printf("\n\nTeclea ENTER para salir, por favor...");
668         fflush(stdin);
669         getchar();
670         exit(-1);
671     }
672     fgets(Aux, 15, Costo);
673     CostoP = atof(Aux);
674     fclose(Costo);
675     return Cant * CostoP;
676 }
677 //-----Calcular Precio de Los Ingredientes-----
678 double CalcularPrecioIngredientes(int Cant)
679 {
680     FILE *Costo;
681     char Aux[15];
682     double CostoP;
683     Costo = fopen("Precios/PrecioIngredientesMarquesitas.txt", "r");
684     if(Costo == NULL)
685     {
686         system("cls");
687         printf("¡¡ERROR!!");
688         printf("\nEL ARCHIVO NO EXISTE...");
689         printf("\n\nTeclea ENTER para salir, por favor...");
690         fflush(stdin);
691         getchar();
692         exit(-1);
693     }
694     fgets(Aux, 15, Costo);
695     CostoP = atof(Aux);
696     return Cant * CostoP;
697 }
```

Cohesión de las funciones

Invocación de funciones:

```
284 {
285     case 1:
286         CantMarquesitas = IngresarMarquesitas();
287         if(CantMarquesitas == 0)
288         {
289             CantIngredientes = 0;
290         }
291         break;
292     case 2:
293         if(CantMarquesitas < 1)
294         {
295             printf("ERROR, DEBE AGREGAR AL MENOS 1 MARQUESITA...");
296             printf("\nTeclea ENTER para continuar, por favor...");
297             fflush(stdin);
298             getchar();
299         }else
300         {
301             CantIngredientes = IngresarIngredientes();
302         }
303         break;
304     case 3:
305         CantEsquites = IngresarEsquites();
306         break;
307     case 4:
308         CantTostiesquites = IngresarTostiesquites();
309         break;
310     case 5:
311         CantToppings = IngresarToppings();
312         break;
313     case 6:
314         CantRefrescos = IngresarRefrescos();
315         break;
316     case 7:
317         CantAguas = IngresarAguas();
318         break;
319     case 8:
320         if(CantMarquesitas == 0)
321         {
322             if(CantEsquites == 0)
323             {
```

Pasó de parámetros:

```
469     for(int i = 1; i <= 7; i++)
470     {
471         switch(i)
472         {
473             case 1:
474                 if(CantMarquesitas > 0)
475                 {
476                     PrecioMarquesitas = CalcularPrecioMarquesitas(CantMarquesitas);
477                     fprintf(NewTicket, "\n%-15s%-5d%10.2lf", "Marquesitas", CantMarquesitas, PrecioMarquesitas);
478                     fprintf(NewTicket, "\n-----");
479                     TOTAL = TOTAL + PrecioMarquesitas;
480                 }
481                 break;
482             case 2:
483                 if(CantIngredientes > 0)
484                 {
485                     PrecioIngredientes = CalcularPrecioIngredientes(CantIngredientes);
486                     fprintf(NewTicket, "\n%-15s%-5d%10.2lf", "Ingredientes", CantIngredientes, PrecioIngredientes);
487                     fprintf(NewTicket, "\n-----");
488                     TOTAL = TOTAL + PrecioIngredientes;
489                 }
490                 break;
491             case 3:
492                 if(CantEsquites > 0)
493                 {
494                     PrecioEsquites = CalcularPrecioEsquites(CantEsquites);
495                     fprintf(NewTicket, "\n%-15s%-5d%10.2lf", "Esquites", CantEsquites, PrecioEsquites);
496                     fprintf(NewTicket, "\n-----");
497                     TOTAL = TOTAL + PrecioEsquites;
498                 }
499                 break;
500             case 4:
501                 if(CantTostiesquites > 0)
502                 {
503                     PrecioTostiesquites = CalcularPrecioTostiesquites(CantTostiesquites);
504                     fprintf(NewTicket, "\n%-15s%-5d%10.2lf", "Tostiesquites", CantTostiesquites, PrecioTostiesquites);
505                     fprintf(NewTicket, "\n-----");
506                     TOTAL = TOTAL + PrecioTostiesquites;
507                 }
508                 break;
509             case 5:
510                 if(CantTostiesquites > 0)
```

Mapecto de requerimientos con las funciones del sistema

| Funciones finales del sistema | | | |
|-------------------------------|----------------------------|---------------|------------|
| Requerimientos Funcionales | | | |
| Código | Descripción breve | Sistema | Porcentaje |
| RF_01 | Validación del usuario | Incluido | 15% |
| RF_02 | Creación de los tickets | Incluido | 10% |
| RF_03 | Cambio de precios | Incluido | 10% |
| RF_04 | Consulta de tickets | Incluido | 10% |
| RF_05 | Ganancias registradas | Incluido | 10% |
| RF_06 | Horario laboral | No Incluido | 5% |
| Requerimientos No Funcionales | | | |
| RNF_01 | Sensibilidad del sistema | Incluido | 10% |
| RNF_02 | Seguridad | Incluido | 10% |
| RNF_03 | Interfaz intuitiva | Incluido | 10% |
| RNF_04 | Accesibilidad del software | Incluido | 10% |
| | | Total: | 95% |

Proceso de Desarrollo

- Como herramienta de código se está utilizando el programa Dev-C++.
- Nos estamos organizando a través de plataformas virtuales y uso de mensajería instantánea para la aclaración de dudas.
- Se está implementando una bitácora para llevar el control del proyecto que sería la siguiente:

| Actividades | Sesión 1 | Sesión 2 | Sesión 3 | Sesión 4 | Sesión 5 | Sesión 6 | Sesión 7 | Sesión 8 | Sesión 9 | Sesión 10 |
|--|----------|----------|----------|----------|----------|----------|----------|----------|----------|-----------|
| Elección del proyecto | | | | | | | | | | |
| Planificación de la estructura del proyecto. | | | | | | | | | | |
| Investigación de antecedentes de la propuesta. | | | | | | | | | | |
| Partición del proyecto (modularización) | | | | | | | | | | |
| Codificación | | | | | | | | | | |
| Revisión de los avances previo a la entrega | | | | | | | | | | |
| Testing | | | | | | | | | | |
| Entrega final | | | | | | | | | | |

Observaciones:

- Las sesiones previas a la **sesión 3**, son fechas previas a la entrega de la propuesta el lunes 13 de abril del 2020.
- Las **sesiones de la 3 a la 7** son fechas entre el 14 y 30 de abril del 2020, siendo el 30 de abril la fecha de entrega de los avances del proyecto.
- Las **sesiones de la 8 a la 10** serían fechas entre el 1 y 11 de mayo del 2020, siendo el 11 de mayo la fecha programada de la entrega final.

Reporte de avance individual de participación en la codificación del proyecto

| Módulo | Alan Josue Barón Pat | Víctor Cauich Davalos |
|---|----------------------|-----------------------|
| Login | | |
| Validación de administrador o trabajador | | |
| Agregar empleado | | |
| Eliminar empleado | | |
| Validación de los productos | | |
| Creación de tickets | | |
| Cambio de precios | | |
| Consulta de tickets | | |
| Visualización de ganancias (día, semana, mes) | | |
| Total: | 40% | 60% |

Reporte de contribución general

| Contribución | Alan Josue Barón Pat | Víctor Cauich Davalos |
|---|----------------------|-----------------------|
| Idea principal | | |
| Antecedentes de la propuesta | | |
| Descripción del software | | |
| Objetivos generales y específicos | | |
| Requerimientos Funcionales y No Funcionales | | |
| Casos de uso | | |
| Diagrama de casos de uso | | |
| Matriz de requerimientos | | |
| Estándar de codificación | | |
| Modularidad | | |
| Proceso de desarrollo | | |
| Bitácoras | | |
| Código | | |
| Documentación del código | | |
| Reporte escrito | | |
| Cartel científico | | |
| Total: | 50% | 50% |