

**LAPORAN HASIL PROYEK AKHIR JURUSAN PENGEMBANGAN PERANGKAT  
LUNAK DAN GIM  
“APLIKASI KEDAI JADUL”**



Disusun Oleh :

NAMA : SALWA SALSABILA  
NO : 26  
NISN : 0096673931  
KELAS : X PPLG 1

**JURUSAN PENGEMBANGAN PERANGKAT LUNAK DAN  
SMK NEGERI 1 KANDEMAN  
TAHUN AJARAN 2025**

## DESKRIPSI PROYEK

Proyek ini bertujuan untuk membuat “**Aplikasi Kedai Jadul**” yang digunakan oleh penjual maupun pembeli jajanan tradisional. Aplikasi ini bertujuan untuk memudahkan penjual maupun pembeli dalam memesan jajanan tradisional melalui platform digital agar lebih cepat dan efektif.

Melalui Aplikasi ini, penjual maupun pembeli dapat :

1. Memudahkan penjual maupun pembeli mencatat transaksi data menu jajanan atau minuman secara cepat dan mudah.
2. Dapat mempermudah pengguna dalam mencatat total perhitungan pembelian secara efektif.
3. Aplikasi ini dibuat guna melatih keterampilan dan pemahaman pengguna melalui Visual Studio Code dengan menggunakan Console C#.

Proyek ini mengimplementasikan konsep dasar pemrograman seperti:

1. Loop (Perulangan)  
Contohnya :
  - a) while (true) digunakan untuk membuat loop pemesanan agar pengguna bisa memasukkan banyak pesanan sampai mengetik “selesai”.
  - b) foreach digunakan untuk menampilkan daftar menu dan ringkasan pesanan.
2. Variabel  
Contohnya :
  - a) double totalHarga digunakan untuk menyimpan total harga pesanan.
  - b) double hargaItem untuk menghitung harga tiap item.
  - c) Variabel jumlah digunakan untuk input jumlah pesanan.
3. Input/Output  
Contohnya :
  - a) Console.WriteLine() untuk menampilkan teks ke layar.
  - b) Console.ReadLine() untuk menerima input dari pengguna.
4. Operator Aritmatika  
Contohnya :
  - a) Perkalian (menu[namaMenu] \* jumlah) untuk menghitung harga item.
  - b) Penjumlahan (totalHarga += hargaItem) untuk menambahkan total harga.
  - c) Diskon (totalHarga \* 0.05) untuk menghitung potongan harga.
5. Percabangan  
Contohnya :

if digunakan untuk:

  - a) Mengecek apakah input “selesai”.
  - b) Mengecek apakah menu ada di Dictionary (menu.ContainsKey).
  - c) Mengecek validasi jumlah pesanan (int.TryParse).
  - d) Mengecek apakah total harga lebih dari 30.000 untuk diskon.
6. Validasi Data

Contohnya :

- a) `int.TryParse()` digunakan untuk memastikan input jumlah adalah angka valid.
- b) Jumlah  $\leq 0$  dicek agar tidak ada pesanan dengan jumlah nol atau negatif.

## 7. Dictionary (Koleksi Key-Value)

Contohnya :

- a) `Dictionary<string, double>` untuk menyimpan **menu** (nama jajanan → harga).
- b) `Dictionary<string, int>` untuk menyimpan **pesanan** (nama jajanan → jumlah).

## RINGKASAN TEORI C# YANG DIGUNAKAN

### Variabel dan Tipe Data

- 1. **Variabel** digunakan untuk menyimpan nilai sementara dalam program.
- 2. Tipe data yang digunakan:
  - a. `string` → teks (nama menu).
  - b. `int` → bilangan bulat (jumlah pesanan).
  - c. `double` → bilangan pecahan (harga dan total).

### Operator Aritmatika

Digunakan untuk perhitungan harga dan diskon.

- a) Perkalian: `menu[namaMenu] * jumlah`.
- b) Penjumlahan: `totalHarga += hargaItem`.
- c) Diskon: `totalHarga * 0.05`.

### Percabangan (If-Else)

- a) Mengecek validasi input (menu ada/tidak, jumlah valid/tidak).
- b) Memberikan diskon jika total harga melebihi batas tertentu.

### Perulangan (Loop)

- a) `while (true)` → menerima pesanan berulang kali sampai pengguna mengetik **"selesai"**.
- b) `foreach` → menampilkan daftar menu dan ringkasan pesanan.

### Input/Output Console

- a) `Console.WriteLine()` → menampilkan teks ke layar.
- b) `Console.ReadLine()` → menerima input dari pengguna.
- c) Digunakan untuk interaksi sederhana antara aplikasi dan pengguna.

**Dictionary** : Koleksi yang menyimpan data dalam bentuk *key-value pair*.

Contoh: Dictionary<string, double> untuk menyimpan menu dan harga. Keunggulan: akses cepat berdasarkan *key* (nama menu).

```
Dictionary<string, double> menu = new Dictionary<string, double>
```

Dictionary<string, double>

**string** → untuk nama menu (contoh: "Onde-onde").

**double** → untuk harga menu (contoh: 2000).

```
Dictionary<string, int> pesanan = new Dictionary<string, int>();
```

Dictionary<string, int>

**string** → nama menu yang dipesan.

**int** → jumlah pesanan.

### Variabel Numerik

```
double totalHarga = 0;
```

**Variabel:** totalHarga

**Tipe Data:** double

Fungsi: menyimpan total harga semua pesanan.

```
double hargaItem = menu[namaMenu] * jumlah;
```

**Variabel:** hargaItem

**Tipe Data:** double

Fungsi: menyimpan harga untuk satu jenis pesanan.

```
int jumlah
```

**Variabel:** jumlah

**Tipe Data:** int Fungsi: menyimpan jumlah item yang dipesan oleh pengguna.

## Variabel String

```
string namaMenu = Console.ReadLine();
```

**Variabel:** namaMenu

**Tipe Data:** string

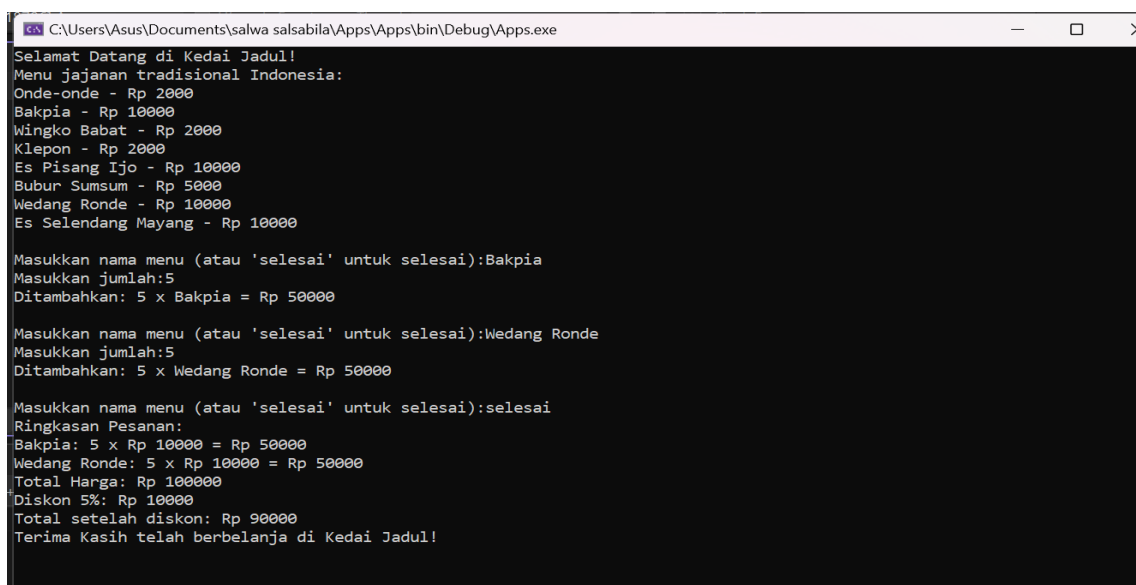
Fungsi: menyimpan input nama menu dari pengguna.

## Input/Output

Console.WriteLine() → tidak menyimpan variabel, tapi menampilkan teks.

Console.ReadLine() → menghasilkan **string** yang kemudian disimpan ke variabel (namaMenu).

## PERANCANGAN PROGRAM



```
C:\Users\Asus\Documents\salsa salsabila\Apps\Apps\bin\Debug\Apps.exe
Selamat Datang di Kedai Jadul!
Menu jajanan tradisional Indonesia:
Onde-onde - Rp 2000
Bakpia - Rp 10000
Wingko Babat - Rp 2000
Klepon - Rp 2000
Es Pisang Ijo - Rp 10000
Bubur Sumsum - Rp 5000
Wedang Ronde - Rp 10000
Es Selendang Mayang - Rp 10000

Masukkan nama menu (atau 'selesai' untuk selesai):Bakpia
Masukkan jumlah:5
Ditambahkan: 5 x Bakpia = Rp 50000

Masukkan nama menu (atau 'selesai' untuk selesai):Wedang Ronde
Masukkan jumlah:5
Ditambahkan: 5 x Wedang Ronde = Rp 50000

Masukkan nama menu (atau 'selesai' untuk selesai):selesai
Ringkasan Pesanan:
Bakpia: 5 x Rp 10000 = Rp 50000
Wedang Ronde: 5 x Rp 10000 = Rp 50000
Total Harga: Rp 100000
Diskon 5%: Rp 10000
Total setelah diskon: Rp 90000
Terima Kasih telah berbelanja di Kedai Jadul!
```

## IMPLEMENTASI PROGRAM

Kode Program Utama

System;

using System.Collections.Generic;

class Program

{

static void Main()

{

Dictionary<string, double> menu = new Dictionary<string, double>

{

{"Onde-onde", 2000 },

{"Bakpia", 10000 },

{"Wingko Babat", 2000 },

{"Klepon", 2000 },

{"Es Pisang Ijo", 10000 },

{"Bubur Sumsum", 5000 },

{"Wedang Ronde", 10000 },

{"Es Selendang Mayang", 10000 },

};

Dictionary<string, int> pesanan = new Dictionary<string, int>();

double totalHarga = 0;

```
Console.WriteLine("=== Selamat Datang di Kedai Jadul ===");

Console.WriteLine("Menu jajanan tradisional Indonesia:");

foreach (var item in menu)

{

    Console.WriteLine($"{item.Key} - Rp {item.Value:N0}");

}

Console.WriteLine();

while (true)

{

    Console.Write("Masukkan nama menu (atau 'selesai' untuk selesai): ");

    string namaMenu = Console.ReadLine();

    if (namaMenu.ToLower() == "selesai")

        break;

    if (!menu.ContainsKey(namaMenu))

    {

        Console.WriteLine("Menu tidak ditemukan. Coba lagi.");

        continue;

    }

    Console.Write("Masukkan jumlah: ");
```

```

if (!int.TryParse(Console.ReadLine(), out int jumlah) || jumlah <= 0)
{
    Console.WriteLine("Jumlah tidak valid. Coba lagi.");
    continue;
}

if (pesanan.ContainsKey(namaMenu))
    pesanan[namaMenu] += jumlah;
else
    pesanan[namaMenu] = jumlah;

double hargaItem = menu[namaMenu] * jumlah;
totalHarga += hargaItem;

Console.WriteLine($"Ditambahkan: {jumlah} x {namaMenu} = Rp
{hargaItem:N0}");
Console.WriteLine();
}

Console.WriteLine("\n=== Ringkasan Pesanan ===");
foreach (var item in pesanan)
{
    double harga = menu[item.Key] * item.Value;

    Console.WriteLine($" {item.Key}: {item.Value} x Rp {menu[item.Key]:N0} = Rp
{harga:N0}");
}

```



```

Console.WriteLine($"Total Harga: Rp {totalHarga:N0}");

if (totalHarga > 30000)
{
    double diskon = totalHarga * 0.05; // 5%
    totalHarga -= diskon;

    Console.WriteLine($"Diskon 5%: Rp {diskon:N0}");
    Console.WriteLine($"Total setelah diskon: Rp {totalHarga:N0}");
}

Console.WriteLine("\nTerima Kasih telah berbelanja di Kedai Jadul!");
Console.ReadLine();

```

#### Penjelasan Kode Program

```

Dictionary<string, double> menu = new Dictionary<string, double>
{
    {"Onde-onde", 2000 },
    {"Bakpia", 10000 },
    {"Wingko Babat", 2000 },
    {"Klepon", 2000 },
    {"Es Pisang Ijo", 10000 },
    {"Bubur Sumsum", 5000 },
    {"Wedang Ronde", 10000 },

```

```
{"Es Selendang Mayang",10000 },
```

#### 1. Deklarasi Variabel

menu adalah nama variabel yang dibuat

type datanya adalah Dictionary<string, double>.

#### 2. Tipe Data Dictionary

Dictionary adalah struktur data koleksi di C# yang menyimpan data dalam bentuk **pasangan key-value**.

string → tipe data untuk **key** (dalam hal ini nama jajanan/minuman).

double → tipe data untuk **value** (dalam hal ini harga jajanan/minuman).

#### 3. Inisialisasi Dictionary

new Dictionary<string, double> membuat objek dictionary baru.

Bagian dalam { . . . } adalah isi dictionary berupa pasangan key-value.

#### 4. Isi Dictionary

Contoh pasangan: {"Onde-onde", 2000 }

"Onde-onde" adalah **key** (nama menu).

2000 adalah **value** (harga menu).

Jadi, setiap item menu tradisional disimpan dengan nama dan harga.

```
Console.WriteLine($"{item.Key} - Rp {item.Value:N0}");
```

#### 1. Console.WriteLine

Digunakan untuk menampilkan teks ke layar (output).

Dalam hal ini, teks yang ditampilkan adalah nama menu dan harga.

#### 2. Interpolasi String (\$"")

Tanda \$ sebelum string memungkinkan kita menyisipkan variabel atau ekspresi langsung ke dalam string dengan {}.

Contoh:

{item.Key} → akan diganti dengan **nama menu** (misalnya "Onde-onde").

{item.Value:N0} → akan diganti dengan **harga menu** (misalnya 2000).

### 3. item.Key dan item.Value

item.Key → adalah **key** dari dictionary (nama jajanan/minuman).

item.Value → adalah **value** dari dictionary (harga jajanan/minuman).

### 4. Format String :N0

N0 adalah format angka dengan **Number format** dan **0 digit desimal**.

Artinya angka akan ditampilkan dengan pemisah ribuan sesuai budaya lokal.

Contoh:

- a. 2000 → ditampilkan sebagai 2,000.
- b. 10000 → ditampilkan sebagai 10,000.

```
Console.Write("Masukkan nama menu (atau 'selesai' untuk selesai): ");
```

```
string namaMenu = Console.ReadLine();
```

```
if (namaMenu.ToLower() == "selesai")
```

```
    break;
```

```
if (!menu.ContainsKey(namaMenu))
```

```
{
```

```
    Console.WriteLine("Menu tidak ditemukan. Coba lagi.");
```

```
    continue;
```

```
}
```

Fungsi: Menampilkan teks ke layar agar pengguna tahu harus memasukkan nama menu.

Console.Write() → menampilkan teks tanpa pindah baris (berbeda dengan () yang otomatis pindah baris).

Variabel: namaMenu bertipe string.

Fungsi: Menyimpan input dari pengguna (nama menu yang diketik di keyboard).

Contoh: jika pengguna mengetik Bakpia, maka namaMenu = "Bakpia".

Fungsi: Mengecek apakah input pengguna adalah "selesai".

ToLower() → mengubah input menjadi huruf kecil semua, sehingga program tetap mengenali meskipun pengguna mengetik "Selesai" atau "SELESAI".

break; → keluar dari loop pemesanan jika pengguna sudah selesai.

Fungsi: Mengecek apakah nama menu yang dimasukkan ada di dalam dictionary menu.

menu.ContainsKey(namaMenu) → mengembalikan true jika menu ada, false jika tidak ada.

Tanda ! berarti **negasi** (tidak). Jadi if (!menu.ContainsKey(namaMenu)) artinya "jika menu tidak ditemukan".

Jika tidak ditemukan:

Program menampilkan pesan "Menu tidak ditemukan. Coba lagi."

continue; → kembali ke awal loop untuk meminta input lagi.

```
Console.Write("Masukkan jumlah: ");
```

```
if (!int.TryParse(Console.ReadLine(), out int jumlah) || jumlah <= 0)
```

```
{
```

```
    Console.WriteLine("Jumlah tidak valid. Coba lagi.");
```

```
    continue;
```

```
}
```

```
if (pesanan.ContainsKey(namaMenu))
```

```
    pesanan[namaMenu] += jumlah;
```

```
else
```

```

        pesanan[namaMenu] = jumlah;

        double hargaItem = menu[namaMenu] * jumlah;

        totalHarga += hargaItem;

        Console.WriteLine($"Ditambahkan: {jumlah} x {namaMenu} = Rp
        {hargaItem:N0}");

        Console.WriteLine();

```

Menampilkan pesan ke layar agar pengguna memasukkan jumlah item yang ingin dipesan.

Console.Write() tidak otomatis pindah baris setelah teks.

Console.ReadLine() → membaca input dari pengguna.

int.TryParse(..., out int jumlah) → mencoba mengubah input menjadi angka (int).

- Jika berhasil, nilai disimpan ke variabel jumlah.
- Jika gagal (misalnya input huruf), hasilnya false.

|| jumlah <= 0 → tambahan validasi agar jumlah tidak boleh nol atau negatif.

Jika input tidak valid → tampilkan pesan error "Jumlah tidak valid. Coba lagi." lalu continue; untuk kembali ke awal loop.

- Mengecek apakah menu yang dipesan sudah ada di dictionary pesanan.
- Jika sudah ada → tambahkan jumlah pesanan (+= jumlah).
- hargaItem → menghitung harga untuk pesanan baru (harga menu × jumlah).
- totalHarga → menambahkan harga pesanan baru ke total keseluruhan.
- Menampilkan ringkasan pesanan yang baru saja ditambahkan.
- {hargaItem:N0} → format angka dengan pemisah ribuan (contoh: 10,000).
- Console.WriteLine(); → menambahkan baris kosong agar output lebih rapi.

```
double harga = menu[item.Key] * item.Value;
```

```

        Console.WriteLine($" {item.Key}: {item.Value} x Rp {menu[item.Key]:N0} = Rp
        {harga:N0}");

    }

```

```

Console.WriteLine($"Total Harga: Rp {totalHarga:N0}");

if (totalHarga > 30000)
{
    double diskon = totalHarga * 0.05; // 5%
    totalHarga -= diskon;

    Console.WriteLine($"Diskon 5%: Rp {diskon:N0}");
    Console.WriteLine($"Total setelah diskon: Rp {totalHarga:N0}");
}

Console.WriteLine("\nTerima Kasih telah berbelanja di Kedai Jadul!");
Console.ReadLine();
}
}

```

**Fungsi:** Menghitung harga untuk setiap item pesanan.

menu[item.Key] → harga satuan dari menu.

item.Value → jumlah pesanan.

harga → hasil perkalian harga satuan × jumlah.

Console.WriteLine(...) → menampilkan ringkasan pesanan, misalnya:

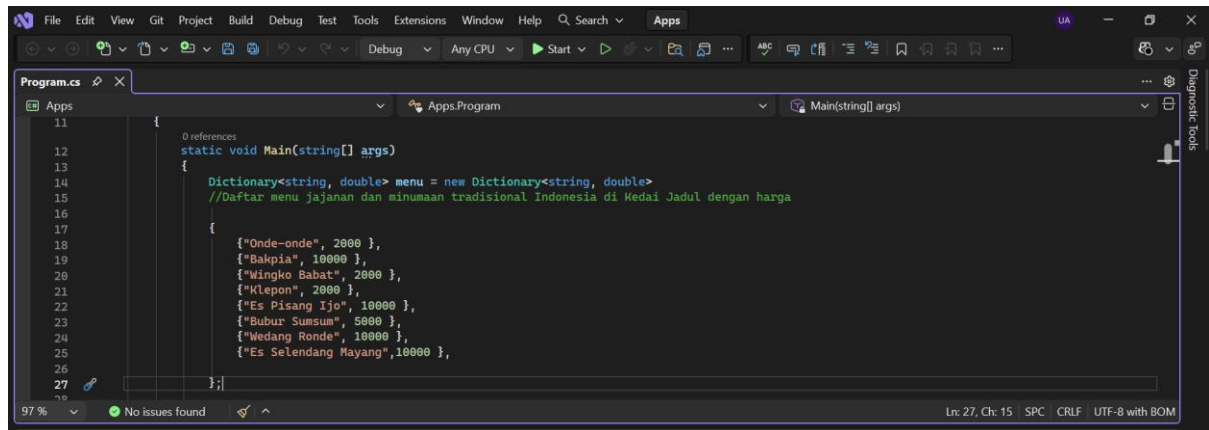
Menampilkan **total harga keseluruhan** dari semua pesanan sebelum diskon.

{totalHarga:N0} → format angka dengan pemisah ribuan.

□ **Fungsi:** Memberikan diskon jika total belanja lebih dari Rp 30.000.

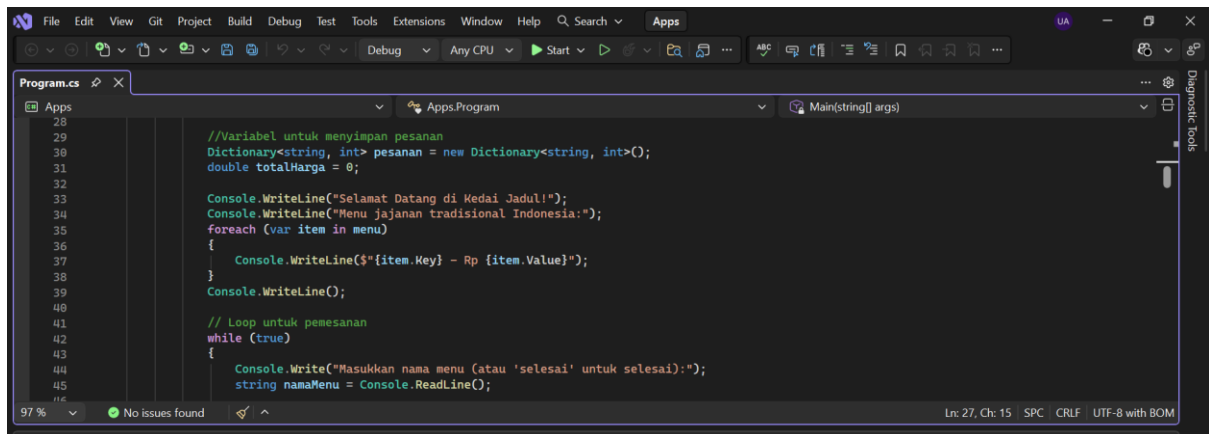
□ double diskon = totalHarga \* 0.05; → menghitung diskon 5%.

- `totalHarga -= diskon;` → mengurangi total harga dengan diskon.
- Menampilkan jumlah diskon dan total harga setelah diskon.
- Menampilkan pesan penutup transaksi.
- `\n` → menambahkan baris kosong sebelum teks agar lebih rapi.
- `Console.ReadLine();` → menunggu input sebelum program selesai, supaya layar tidak langsung tertutup.



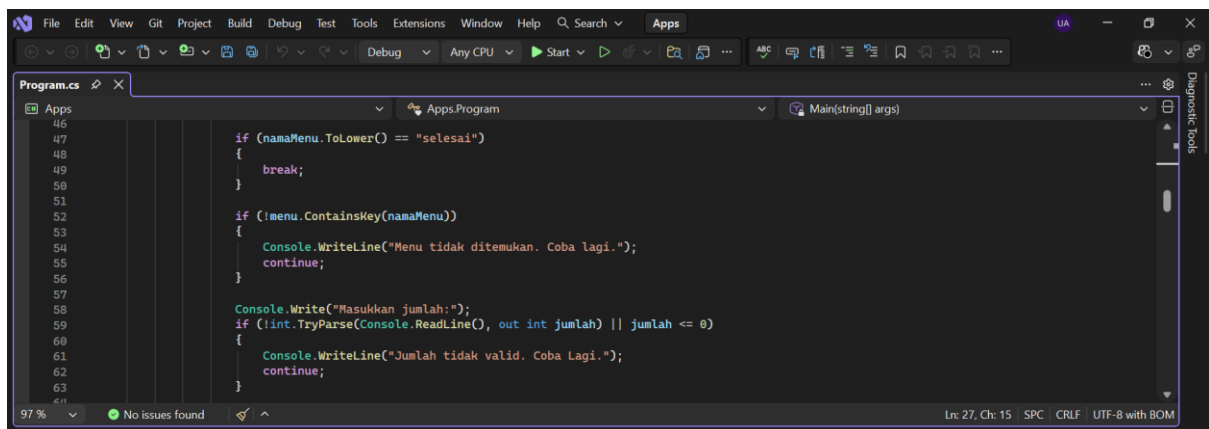
This screenshot shows the first part of the C# code in Program.cs. It defines a static Main method that takes an array of arguments. Inside the method, a Dictionary<string, double> named 'menu' is created. A comment indicates that this dictionary contains traditional Indonesian food and drink prices. The dictionary is populated with several items, including 'Onde-onde', 'Bakpia', 'Klepon', 'Es Pisang Ijo', 'Bubur Sumsum', 'Wedang Ronde', and 'Es Selendang Mayang', each with its corresponding price.

```
11 {
12     0 references
13     static void Main(string[] args)
14     {
15         Dictionary<string, double> menu = new Dictionary<string, double>
16         //Daftar menu jajanan dan minuman tradisional Indonesia di Kedai Jadul dengan harga
17         {
18             {"Onde-onde", 2000 },
19             {"Bakpia", 10000 },
20             {"Klepon", 2000 },
21             {"Es Pisang Ijo", 10000 },
22             {"Bubur Sumsum", 5000 },
23             {"Wedang Ronde", 10000 },
24             {"Es Selendang Mayang", 10000 },
25         }
26     }
27 }
```



This screenshot shows the middle section of the C# code. It includes comments for variable declarations and the logic for displaying the menu. The code uses Console.WriteLine to show a welcome message and the menu items. A foreach loop iterates through the 'menu' dictionary to display each item and its price. After displaying the menu, a while loop starts, which prompts the user to enter a menu item or 'selesai' to finish. The user's input is stored in the 'namaMenu' variable.

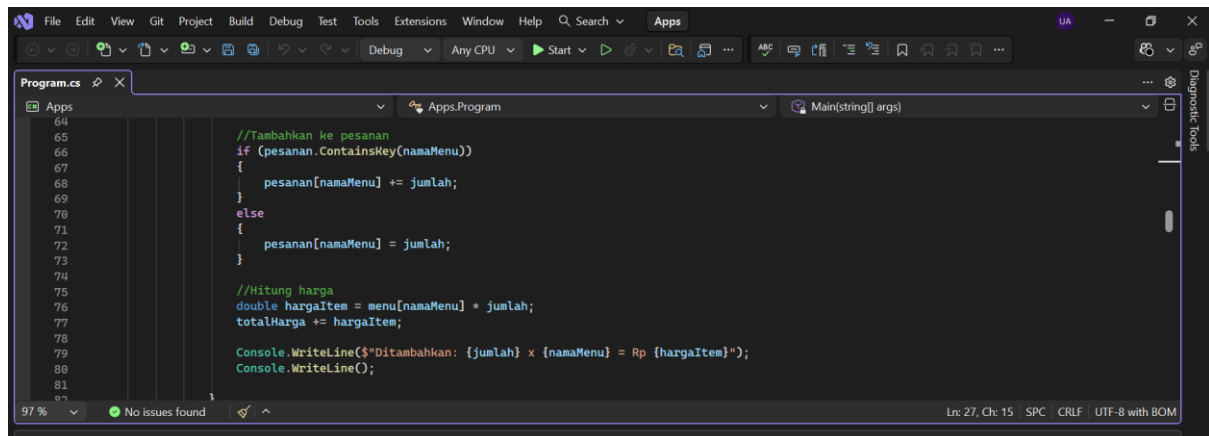
```
28 //Variabel untuk menyimpan pesanan
29 Dictionary<string, int> pesanan = new Dictionary<string, int>();
30 double totalHarga = 0;
31
32
33 Console.WriteLine("Selamat Datang di Kedai Jadul!");
34 Console.WriteLine("Menu jajanan tradisional Indonesia:");
35 foreach (var item in menu)
36 {
37     Console.WriteLine($"{item.Key} - Rp {item.Value}");
38 }
39 Console.WriteLine();
40
41 // Loop untuk pemesanan
42 while (true)
43 {
44     Console.Write("Masukkan nama menu (atau 'selesai' untuk selesai):");
45     string namaMenu = Console.ReadLine();
46 }
```



This screenshot shows the final part of the C# code, which handles user input validation. It checks if the user entered 'selesai' (converted to lowercase) to break the loop. If the entered menu name is not found in the dictionary, it prompts the user to try again. Additionally, it prompts the user to enter a quantity ('jumlah') and checks if it's a valid integer. If the input is invalid, it prompts the user to try again.

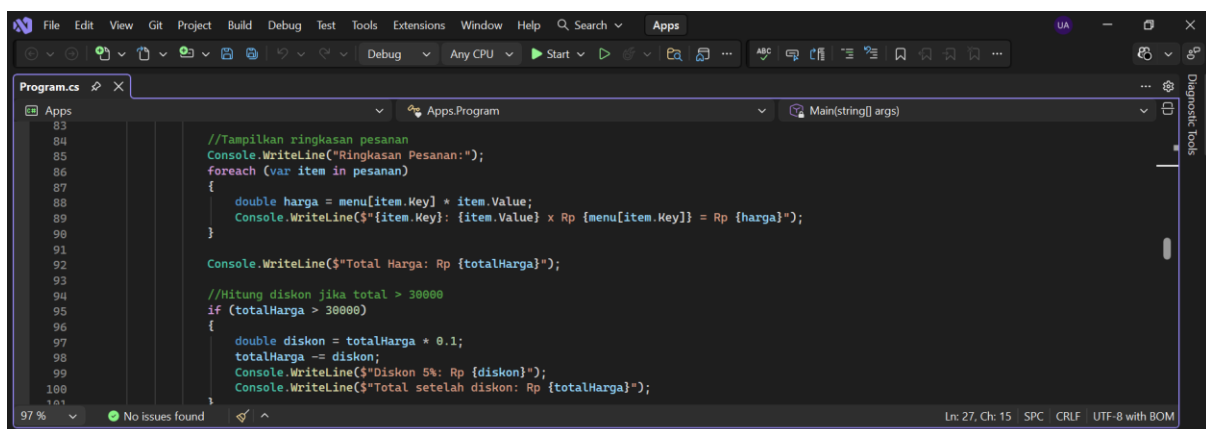
```
47 if (namaMenu.ToLower() == "selesai")
48 {
49     break;
50 }
51
52 if (!menu.ContainsKey(namaMenu))
53 {
54     Console.WriteLine("Menu tidak ditemukan. Coba lagi.");
55     continue;
56 }
57
58 Console.Write("Masukkan jumlah:");
59 if (!int.TryParse(Console.ReadLine(), out int jumlah) || jumlah <= 0)
60 {
61     Console.WriteLine("Jumlah tidak valid. Coba Lagi.");
62     continue;
63 }
```





This screenshot shows a Visual Studio Code editor window with a C# file named Program.cs. The code is part of a console application named Apps, with the current method being Main(string[] args). The code logic is as follows: it checks if a menu item already exists in the 'pesanan' dictionary. If it does, it increments the quantity ('jumlah'). If it doesn't, it adds the item with a quantity of 1. Then, it calculates the item price ('hargaItem') by multiplying the menu price ('menu[namaMenu]') by the quantity, and adds it to the 'totalHarga'. Finally, it prints a confirmation message to the console: "Ditambahkan: {jumlah} x {namaMenu} = Rp {hargaItem}".

```
64
65
66 //Tambahkan ke pesanan
67 if (pesanan.ContainsKey(namaMenu))
68 {
69     pesanan[namaMenu] += jumlah;
70 }
71 else
72 {
73     pesanan[namaMenu] = jumlah;
74 }
75
76 //Hitung harga
77 double hargaItem = menu[namaMenu] * jumlah;
78 totalHarga += hargaItem;
79
80 Console.WriteLine($"Ditambahkan: {jumlah} x {namaMenu} = Rp {hargaItem}");
81 Console.WriteLine();
```



This screenshot shows the continuation of the C# code in Program.cs. It starts by printing a separator line "Ringkasan Pesanan:". Then, it iterates through the 'pesanan' dictionary to display each item's name, quantity, and price, formatted as "item.Key: {item.Value} x Rp {menu[item.Key]} = Rp {harga}". After listing all items, it prints the total price: "Total Harga: Rp {totalHarga}". Next, it checks if the total price is greater than 30,000. If so, it calculates a 10% discount ('diskon'), subtracts it from the total, and prints the discounted total: "Diskon 5%: Rp {diskon}" and "Total setelah diskon: Rp {totalHarga}".

```
83
84 //Tampilkan ringkasan pesanan
85 Console.WriteLine("Ringkasan Pesanan:");
86 foreach (var item in pesanan)
87 {
88     double harga = menu[item.Key] * item.Value;
89     Console.WriteLine($"{item.Key}: {item.Value} x Rp {menu[item.Key]} = Rp {harga}");
90 }
91
92 Console.WriteLine($"Total Harga: Rp {totalHarga}");
93
94 //Hitung diskon jika total > 30000
95 if (totalHarga > 30000)
96 {
97     double diskon = totalHarga * 0.1;
98     totalHarga -= diskon;
99     Console.WriteLine($"Diskon 5%: Rp {diskon}");
100     Console.WriteLine($"Total setelah diskon: Rp {totalHarga}");
101 }
```

## KESIMPULAN :

Dari proyek akhir yang saya buat dapat disimpulkan bahwa Aplikasi sederhana Kedai Jadul yang saya buat memiliki tujuan untuk mempermudah pengguna dalam memesan jajanan tradisional dengan mudah hanya melalui handphone saja tanpa perlu mengantri lama-lama untuk mememesannya. Aplikasi ini saya buat guna melatih kemampuan saya dalam membuat coding.





