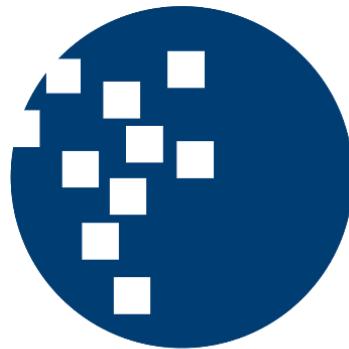


PROJECT DATABASE SYSTEM
PERANCANGAN APLIKASI APOTEK KIMIA FARMA



Disusun oleh :

Dava Virgio Kertawijaya (00000056848)
Nigel Andrian (00000055946)
Michael Owen Kohar (00000056755)
Samuel Andrew (00000056975)
Rifcki Dwiyansyah (00000055611)
Fathimah Az Zahra (00000056188)
Willyam Louise Vernando (00000055805)

Program Studi: Information System
Mata Kuliah: UAS - IS 431 Visual Programming

UNIVERSITAS MULTIMEDIA NUSANTARA
TANGERANG
SEMESTER GENAP 2021/2022

HALAMAN PENGESAHAN DOKUMEN

Dokumen ini telah disetujui sebagai Laporan Akhir untuk penggerjaan Database System Perancangan Aplikasi apotek sederhana untuk membantu proses *procurement* dan *fulfillment* dan telah mencerminkan pemahaman akan kebutuhan di apotek-apotek yang akan menjadi pengguna akhir dari dokumen ini.

Disusun oleh tim *Enterprise Architecture*

NIM	Nama	Program	Tanda Tangan
00000056848	Dava Virgio Kertawijaya	Sistem Informasi	
00000055946	Nigel Andrian	Sistem Informasi	
00000056755	Michael Owen Kohar	Sistem Informasi	
00000056975	Samuel Andrew	Sistem Informasi	
00000055611	Rifcki Dwiyansyah	Sistem Informasi	
00000056188	Fathimah Az Zahra	Sistem Informasi	
00000055805	Willyam Louise Vernando	Sistem Informasi	

Topik	:	PERANCANGAN APLIKASI APOTEK SEDERHANA (GUDANG, TRANSAKSI KASIR, DAN DATA EXPIRED OBAT)
--------------	----------	---

Judul Project (Indonesia)	:	PERANCANGAN APLIKASI APOTEK KIMIA FARMA
Judul Project (Inggris)	:	PHARMACY CHEMICAL APPLICATION DESIGN
Pembimbing	:	1. L00079 - Agus Sulaiman 2. 074887 - Jansen Wiratama

Dibimbing Oleh:

Jansen Wiratama

Dosen Sistem Informasi

Agus Sulaiman

Dosen Sistem Informasi

Tanggal:

Daftar Isi

BAB I	6
PENDAHULUAN	6
1.1. Latar Belakang	6
1.2. Rumusan Masalah	7
1.3. Tujuan	7
1.4. Profil Perusahaan	8
1.5. Proses Aplikasi	8
BAB II	9
ANALISIS MASALAH	9
2.1. Masalah Bisnis	9
2.2. System Requirement	10
2.2.1. Functional Requirements	10
2.2.2. Non-Functional Requirements	10
2.3 Use Case Diagram	16
BAB III	17
PENYELESAIAN MASALAH	17
3.1. Proses Normalization	17
3.1.1. UNF (Unnormalized Form)	17
3.1.2. 1NF	17
3.1.3. 2NF	17
3.1.4. 3NF	18
3.2. Entity Relationship Modeling	18
3.3. Enhanced Entity Relationship Diagram	19
BAB IV	23
PENYELESAIAN MASALAH	23
4.1. Tabel Relasi	23
4.2. Penjelasan & Gambar User Interface Aplikasi & Kodingan	23
4.2.1. Mdi Form	23
4.2.2. Login	26
4.2.3. Medicine	29
4.2.4. Supplier	33
4.2.5. Customer	37
4.2.6. Employee	41
4.2.7. Transaksi	46
4.2.8. Utility	50
4.3. Query Pembuatan Tabel Basis Data	51

4.3.1. Create dan Use Database	51
4.3.2. Create Table Employee	51
4.3.3. Create Table Customer	52
4.3.4. Create Table Supplier	52
4.3.5. Create Table TransactionHeader	52
4.3.6. Create Table Medicine	52
4.3.7. Create Table TransactionDetail	53
BAB V	54
KESIMPULAN	54
5.1. Kesimpulan	54
Peran Anggota	55

BAB I

PENDAHULUAN

1.1. Latar Belakang

PT. Kimia farma yang menyediakan produk berupa obat - obatan bagi berbagai macam penyakit, suplemen kesehatan, dan obat - obatan lainnya. Kesalahan teknis sering kali terjadi, seperti contohnya salah menginput nama dan jumlah obat - obatan. Masalah paling sering terjadi karena sistem. Karena kesalahan tersebut obat - obatan yang tersimpan di gudang akan ikut terpengaruh dan bisa saja penyimpanan gudang terbengkalai. Maka dari itu kami memutuskan untuk membuat project aplikasi untuk membantu memudahkan pemahaman terdapat data. Pengaruh utama umat manusia di zaman ini adalah memahami teknologi. Kemajuan teknologi yang tidak bisa dihindarkan, yang bertujuan untuk membuka mata manusia agar mau memahami teknologi lebih dalam karena teknologi yang nantinya akan menolong kita untuk menjalani aktifitas sehari - harinya. Kami berencana untuk membuat project pembuatan aplikasi apotek PT. Kimia farma yang menggunakan database SQL server sebagai penyimpan data apotek. Aplikasi yang nantinya akan mendukung kegiatan penginputan data sehingga dapat dicari secara mudah melewati aplikasi tersebut dengan dukungan database. Permasalahan yang umum ditemukan pada yang bukan Gen Z yaitu sulit mengerti tentang teknologi, kami membuat project ini guna memudahkan pengguna manapun dan tidak terbatas umur apapun untuk mengerti mudahkan memakai aplikasi yang nantinya kami buat.

Oleh karena itu banyak orang membutuhkan aplikasi yang menolong mereka untuk melihat data dengan mudah. Project aplikasi yang kami rancang berfokus pada data obat - obatan di PT. Kimia farma. Pada nantinya data yang telah dibuat menggunakan SQL Server tersebut akan digabungkan pada bahasa C# untuk dibuat sebagai aplikasi. Kami menjaga akurasi penginputan data kami untuk menghindari kesalahpahaman data.

Berdasarkan penjelasan yang telah kami jelaskan, maka kelompok kami memutuskan untuk merancang aplikasi berbasis Desktop untuk menginput data

apotek yang dituju. Kami harapkan aplikasi ini dapat membantu pemakai dalam melihat pemasukan data setiap barang tersebut datang dan pergi. Data yang dituliskan pada table yang dibuat memakai database akan selalu diperbaharui bila ada data yang berubah - ubah setiap saatnya.

1.2. Rumusan Masalah

- Bagaimana cara merancang sistem informasi yang dapat mengelola produk pada gudang ?
- Bagaimana cara menambahkan fitur input data pada sistem informasi yang akan dirancang ?
- Bagaimana cara mempermudah proses obat - obatan melalui sistem informasi ?
- Bagaimana cara mengatasi produk yang hampir habis masa waktunya ?
- Bagaimana cara merubah data yang diinput dengan baik ?
- Bagaimana cara mengetahui produk yang jarang dicari pemakai dan di jual di berbagai apotik ?
- Bagaimana cara yang mudah dimengerti antara personil dan juga pengurus obat - obatan ?

1.3. Tujuan

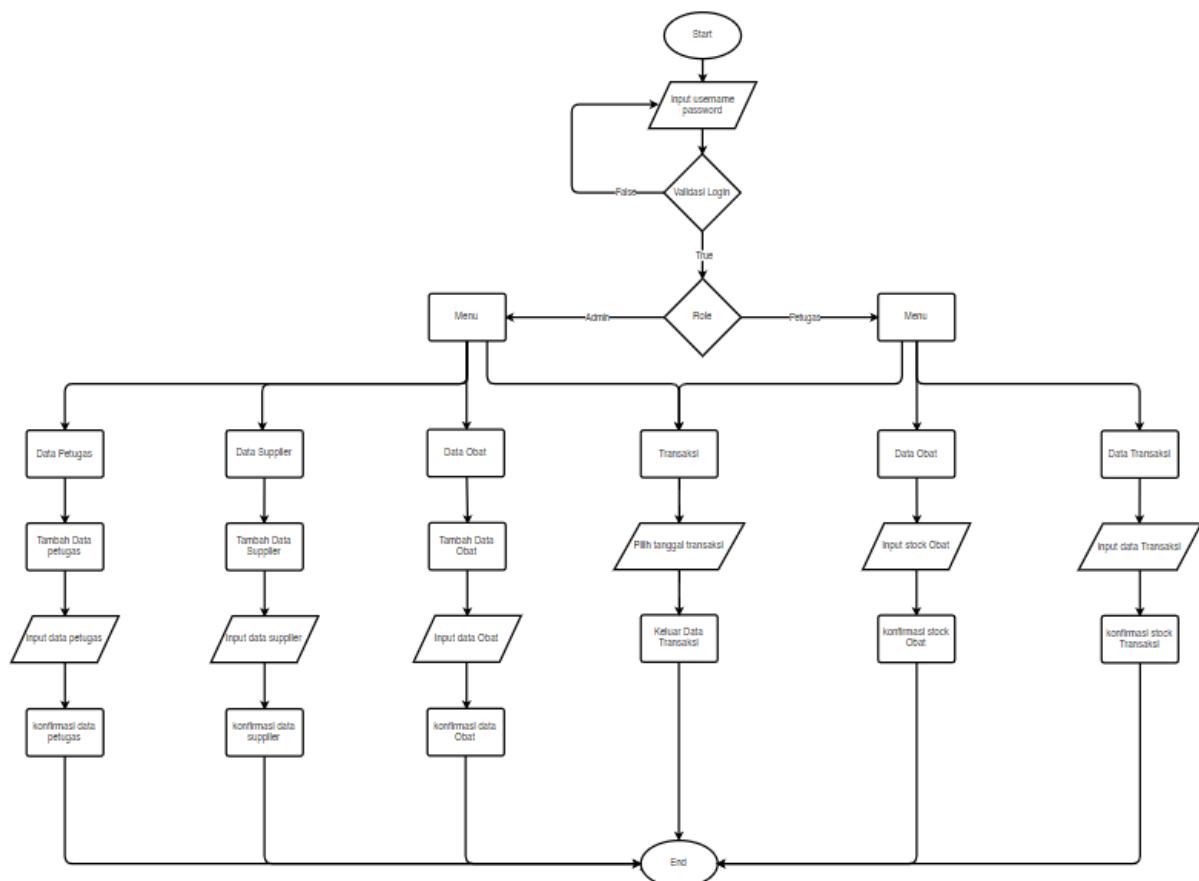
- Untuk membantu apotek kimia farma dalam mengurus bisnis apoteknya yang akan memiliki cabang - cabang di semua penjuru wilayah.
- Membantu pengolahan data yang masuk pada kasir. Termasuk menghitung jumlah uang yang harus dibayar oleh pelanggan.
- Melihat beberapa stok barang yang berada di gudang. Hal tersebut dapat menghindari terjadi kehabisan stok barang.
- Membuat fitur kasir yang akan mempermudah kasir untuk menginput data

1.4. Profil Perusahaan

Aplikasi Apotek Kimia Farma adalah aplikasi yang digunakan untuk melakukan pengecekan oleh pegawai Kimia Farma, dimana saat menggunakan aplikasi pegawai dapat melihat data obat yang masih tersedia dan juga memudahkan pekerja kasir.

Dari aplikasi Apotek Kimia Farma pengguna dapat melihat dan mengetahui stok obat yang masih tersisa di gudang ada berapa banyak dan saat pelanggan ingin menanyakan stok obat tersebut, pegawai dapat dengan mudah mengecek dengan data yang real-time. Tujuan dari aplikasi Apotek Kimia Farma adalah memberikan informasi yang real-time agar pegawai mendapatkan informasi yang mereka perlu untuk mengetahui stok obat.

1.5. Proses Aplikasi



BAB II

ANALISIS MASALAH

2.1. Masalah Bisnis

Di masa pandemi ini, proses bisnis suatu perusahaan mengalami perubahan signifikan yang disebabkan perkembangan teknologi. Perubahan sistem konvensional sampai menjadi sistem berbasis teknologi dilakukan dengan tujuan pengelolaan data dan pengoperasian yang lebih praktis dan hemat waktu. Penerapan sistem baru ini benar-benar merombak cara kerja suatu proses bisnis. Contohnya, seperti PT. Kimia Farma yang menyediakan produk berupa obat-obatan bagi berbagai macam penyakit, suplemen kesehatan, dan barang lainnya. Dengan menggunakan sistem berbasis teknologi berupa aplikasi, kesalahan teknis yang mungkin terjadi saat menggunakan sistem konvensional dapat diminimalisir.

Masalah yang mungkin timbul dengan menggunakan sistem konvensional, yaitu kualitas barang yang semakin menurun di dalam penyimpanan gudang. Bahkan, produk yang disimpan bisa sampai mengalami kadaluarsa. Oleh karena itu, kami memutuskan untuk membuat aplikasi penyimpanan gudang dan penjualan produk, serta pengurutan stok di gudang penyimpanan berdasarkan tanggal produk pertama kali di input atau secara First In First Out (FIFO) yang akan membantu para personil apoteker untuk mengetahui jumlah, dan tanggal kadaluarsa stok barang. Selain itu, kami juga menambahkan sistem pemberitahuan saat stok barang yang terdapat di dalam penyimpanan gudang mencapai jumlah tertentu menggunakan kode warna merah pada produk produk tertentu. Dengan demikian, para personil dapat dengan mudah melakukan stok logistik dan pengecekan jumlah stok yang tersedia agar dapat melakukan restock barang dengan cepat kepada supplier. Aplikasi ini juga dapat digunakan untuk melakukan transaksi penjualan kepada pembeli. Sehingga, pembeli tidak perlu pergi ke apotek secara langsung. Melainkan, hanya memesan produk melalui aplikasi.

2.2. System Requirement

2.2.1. Functional Requirements

- Melakukan *Log in* dan menerima informasi transaksi yang akan dilakukan
- Memperoleh informasi berupa *transaction_id*
- Menginput daftar obat yang dipesan oleh pelanggan
- Melakukan perhitungan total dari harga obat yang dipesan
- Menampilkan harga yang harus dibayar oleh pelanggan
- Memperbarui status pembayaran dari pending menjadi complete ketika pembayaran sudah dilakukan
- Menyimpan data transaksi dan mengurangi stok sesuai dengan jumlah yang ada di transaksi
- Terdapat berbagai informasi yang harus dihasilkan oleh sistem, antara lain:
 - Data Nama Obat
 - Data Jumlah Obat
 - Data Harga /Obat
 - Data Total Harga
 - Data Jumlah Stock Obat Yang Tersedia
 - Data Nama Stock Obat Yang Berkurang

2.2.2. Non-Functional Requirements

1. Manageability

Title	Requirements
Availability	Database Kimia Farma akan aktif selama 24 jam dan tidak memiliki <i>downtime</i> , kecuali pada saat melakukan pemeliharaan (<i>maintenance</i>).
Recoverability	Database Kimia Farma akan dipulihkan dalam waktu 30 menit apabila terjadi <i>downtime</i> yang tidak terencana.

Title	Requirements
Maintainability	Setiap tampilan pada aplikasi Kimia Farma akan diberikan tombol kembali (<i>back</i>) untuk memudahkan <i>user</i> berpindah dari menu satu menuju menu lainnya.
Supportability	Aplikasi Kimia Farma harus memberikan informasi secara terperinci tentang kondisi kesalahan beserta sumbernya sehingga perusahaan dapat mengidentifikasi dan mengatasi permasalahan secara efisien.
Data Integrity	Perusahaan harus memastikan bahwa setiap informasi yang telah ditulis atau diberikan oleh <i>user</i> telah tersimpan dalam database.
Process Notification	<i>Process notification</i> mampu memperkirakan sisa estimasi waktu yang dibutuhkan dalam suatu proses. Apabila sisa estimasi waktu tidak memungkinkan, maka aplikasi tidak akan menampilkan <i>process notification</i> .
Confirmation Box	Setiap kali <i>user</i> melakukan aksi yang dapat membawa pengaruh pada keputusan terhadap data. Maka, reaksi tersebut memicu munculnya <i>confirmation box</i> pada layar untuk memastikan bahwa aksi <i>user</i> telah dilakukan dengan benar.

Title	Requirements
Monitoring Report	Perusahaan memastikan bahwa tidak ada kesalahan yang muncul dalam mengawasi atau memonitor setiap data yang tersimpan dalam database.

2. Security

Title	Requirements
Authentication	Sistem hanya dapat digunakan oleh admin dan petugas kasir.
Authorization	Admin dapat mengakses data transaksi serta gudang yang dioperasikan dan pembayaran (<i>payment</i>). Petugas kasir dapat mengakses data warehouse.
Compliance	Aplikasi akan dijalankan dengan sistem <i>firewall</i> .

3. Performance

Title	Requirements
Responsiveness	Berikut terdapat <i>performance load time</i> yang diharapkan dalam aplikasi Kimia Farma: <ul style="list-style-type: none"> - Maksimal 3 detik untuk memasukkan data pemesanan. - Maksimal 20 detik untuk melakukan view pada data pemesanan.

	<ul style="list-style-type: none"> Maksimal 20 detik untuk melakukan view pada data pembayaran. Maksimal 20 detik untuk melakukan view pada data tracking pemesanan.
Concurrency	Jumlah maksimum <i>concurrent user</i> adalah 10 <i>user</i> .
Resource Usage	Aplikasi Kimia Farma harus dirancang untuk menggunakan sumber daya berupa perangkat keras (<i>hardware</i>), baik di sisi klien maupun sisi server.
Agility	Aplikasi Kimia Farma harus bisa menangani dan memastikan akan integritas data apabila terjadi kondisi <i>overload</i> pada data yang ada.

4. User Interface Requirement

Title	Requirements
Ease of Use	Sistem harus memiliki tampilan yang bersifat <i>user-friendly</i> . Tampilan harus mewakili fungsi tersendiri secara efisien yang dapat dimengerti dan tidak menimbulkan kebingungan bagi pengguna.
Attractiveness	Sistem harus memiliki tampilan semenarik mungkin dengan tujuan agar pengguna dapat mengetahui umpan balik (<i>feedback</i>) dari tindakan yang telah dilakukan terhadap sistem melalui respon yang diberikan oleh sistem.

Learn Ability	<i>User</i> seperti admin dan petugas kasir harus memiliki pengetahuan dasar dalam menggunakan aplikasi dengan benar melalui pelatihan selama satu hari.
---------------	--

5. Interoperability

Title	Requirements
Interoperability	Aplikasi sebaiknya dijalankan pada <i>hardware</i> berupa <i>desktop</i> dengan spesifikasi ruang memori 64GB, RAM 4GB, serta koneksi internet melalui data seluler maupun wifi untuk menjamin kelancaran transmisi data.

6. Backups

Title	Requirements
Backups	Cadangan data (<i>backup data</i>) akan tersimpan dalam memori yang berfungsi untuk melakukan <i>backup</i> data yang bersifat penting. Setiap data yang dimasukkan akan secara otomatis tersimpan ke dalam memori cadangan.
Data Protection	Data <i>protection</i> dilakukan dengan sistem <i>firewall</i> yang aktif setiap saat dan dibutuhkan proses verifikasi oleh pihak yang berwenang setiap kali ada

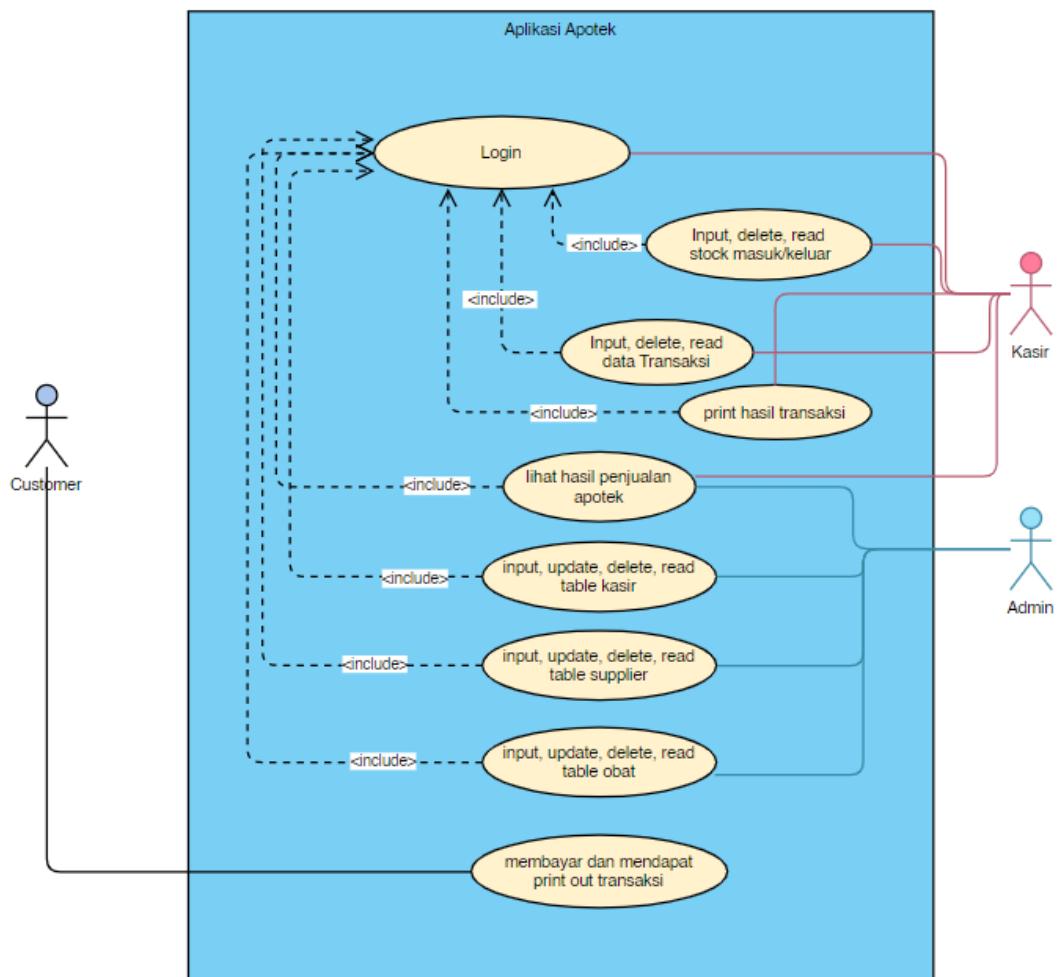
	<i>user</i> yang mencoba untuk melakukan perubahan pada data yang ada.
Availability	Data <i>backup</i> yang berada dalam memori cadangan akan selalu diekspor ke server utama (<i>main server</i>).
Disaster Recovery	Data <i>backup</i> yang berada dalam memori cadangan dapat diekspor ke <i>main server</i> apabila terjadi peristiwa kerusakan data, serangan <i>cyber</i> , bencana alam, <i>human error</i> (kesalahan disebabkan oleh manusia), kerusakan <i>hardware</i> yang digunakan pada <i>main server</i> , dan peristiwa lainnya yang tidak terduga.

7. Infrastructure Requirements

Title	Requirements
Clients	Perangkat yang digunakan untuk servers harus memiliki spesifikasi ruang memori 64GB, RAM 2GB. Sistem operasi yang digunakan adalah Windows 10 dengan <i>processor</i> Intel I3 gen 7.
Servers	Perangkat yang digunakan untuk servers harus memiliki spesifikasi ruang memori 64GB, RAM 2GB. Sistem operasi yang digunakan adalah Windows 10 dengan minimum prosesor Intel i5 gen 7.

Network	Koneksi internet disertai dengan wifi yang dihubungkan pada setiap perangkat. Bandwidth yang akan digunakan sebesar 10 Mbit/s dengan kecepatan sebesar 1 Gbps.
Peripheral	Hardware yang diperlukan untuk mengoperasikan aplikasi ini adalah perangkat Desktop dan wifi untuk koneksi internet.

2.3 Use Case Diagram



BAB III

PENYELESAIAN MASALAH

3.1. Proses Normalization

3.1.1. UNF (*Unnormalized Form*)

UNF									
	TransactionID	TransactionDate	CustID	CustName	CustAddress	CustPhone	MedicineID	MedicineName	
1	T0001	25-Apr-2022	C0001	Willyam	Gading Serpong, Tangerang	0123-4567-8910	M0001	Inhaler	
2							M0002	Utrogestan	
3							M0003	Supralisin	
4							M0004	Tremenza	
5							M0005	Favipiravir	
6									
7									
8									
9									
10									
MedicineQty	MedicinePrice	MedicineDate		Qty	Subtotal	Total	EmployeeID	EmployeeName	
11	100	IDR 50.000	22-Juli-2022						
12	20	IDR 30.000	23-Augustus-2022						
13	20	IDR 50.000	15-Oktober-2022						
14	50	IDR 50.000	8-September-2022						
15	10	IDR 60.000	17-November-2022						
16									
17	UserName	Password	Email	Date Join	EmployeeAddress	SupplierID	SupplierName	SupplierAddress	SupplierPhone
18									
19	Louise	database_04	louise123@gmail.com	22-Apr-2022	BSD Alam Sutra, Tangerang	SP0001	PT. Kimia Farma	Bouverland Barat Raya, Jakarta Utara	021-7759-4360
20									
21									
22									
23									
24									

3.1.2. 1NF

1NF								
	TransactionID	TransactionDate	CustID	CustName	CustAddress	CustPhone	MedicineID	MedicineName
25	T0001	25-Apr-2022	C0001	Willyam	Gading Serpong, Tangerang	0123-4567-8910	M0001	Inhaler
26	T0001	25-Apr-2022	C0001	Willyam	Gading Serpong, Tangerang	0123-4567-8910	M0002	Utrogestan
27	T0001	25-Apr-2022	C0001	Willyam	Gading Serpong, Tangerang	0123-4567-8910	M0003	Supralisin
28	T0001	25-Apr-2022	C0001	Willyam	Gading Serpong, Tangerang	0123-4567-8910	M0004	Tremenza
29	T0001	25-Apr-2022	C0001	Willyam	Gading Serpong, Tangerang	0123-4567-8910	M0005	Favipiravir
30	T0001	25-Apr-2022	C0001	Willyam	Gading Serpong, Tangerang	0123-4567-8910		
31								
MedicineQty	MedicinePrice	MedicineDate		Qty	Subtotal	Total	EmployeeID	EmployeeName
32	100	IDR 50.000	22-Juli-2022	200	IDR 9.700.000	IDR 9.700.000	EM0001	Farhan Korapat
33	20	IDR 30.000	23-Augustus-2022	200	IDR 9.700.000	IDR 9.700.000	EM0001	Farhan Korapat
34	20	IDR 50.000	15-Oktober-2022	200	IDR 9.700.000	IDR 9.700.000	EM0001	Farhan Korapat
35	50	IDR 50.000	8-September-2022	200	IDR 9.700.000	IDR 9.700.000	EM0001	Farhan Korapat
36	10	IDR 60.000	17-November-2022	200	IDR 9.700.000	IDR 9.700.000	EM0001	Farhan Korapat
37								
EmployeeAddress	SupplierID	SupplierName	SupplierAddress	SupplierPhone				
38	BSD Alam Sutra, Tangerang	SP0001	PT. Kimia Farma	bouverland Barat Raya, Jakarta Utara	021-7759-4360			
39	BSD Alam Sutra, Tangerang	SP0001	PT. Kimia Farma	bouverland Barat Raya, Jakarta Utara	021-7759-4360			
40	BSD Alam Sutra, Tangerang	SP0001	PT. Kimia Farma	bouverland Barat Raya, Jakarta Utara	021-7759-4360			
41	BSD Alam Sutra, Tangerang	SP0001	PT. Kimia Farma	bouverland Barat Raya, Jakarta Utara	021-7759-4360			
42	BSD Alam Sutra, Tangerang	SP0001	PT. Kimia Farma	bouverland Barat Raya, Jakarta Utara	021-7759-4360			
43	BSD Alam Sutra, Tangerang	SP0001	PT. Kimia Farma	bouverland Barat Raya, Jakarta Utara	021-7759-4360			
44	BSD Alam Sutra, Tangerang	SP0001	PT. Kimia Farma	bouverland Barat Raya, Jakarta Utara	021-7759-4360			
45								
UserName	Password	Email	Date Join					
46	Louise	database_04	louise123@gmail.com	22-Apr-2022				
47	Louise	database_04	louise123@gmail.com	22-Apr-2022				
48	Louise	database_04	louise123@gmail.com	22-Apr-2022				
49	Louise	database_04	louise123@gmail.com	22-Apr-2022				
50	Louise	database_04	louise123@gmail.com	22-Apr-2022				
51	Louise	database_04	louise123@gmail.com	22-Apr-2022				

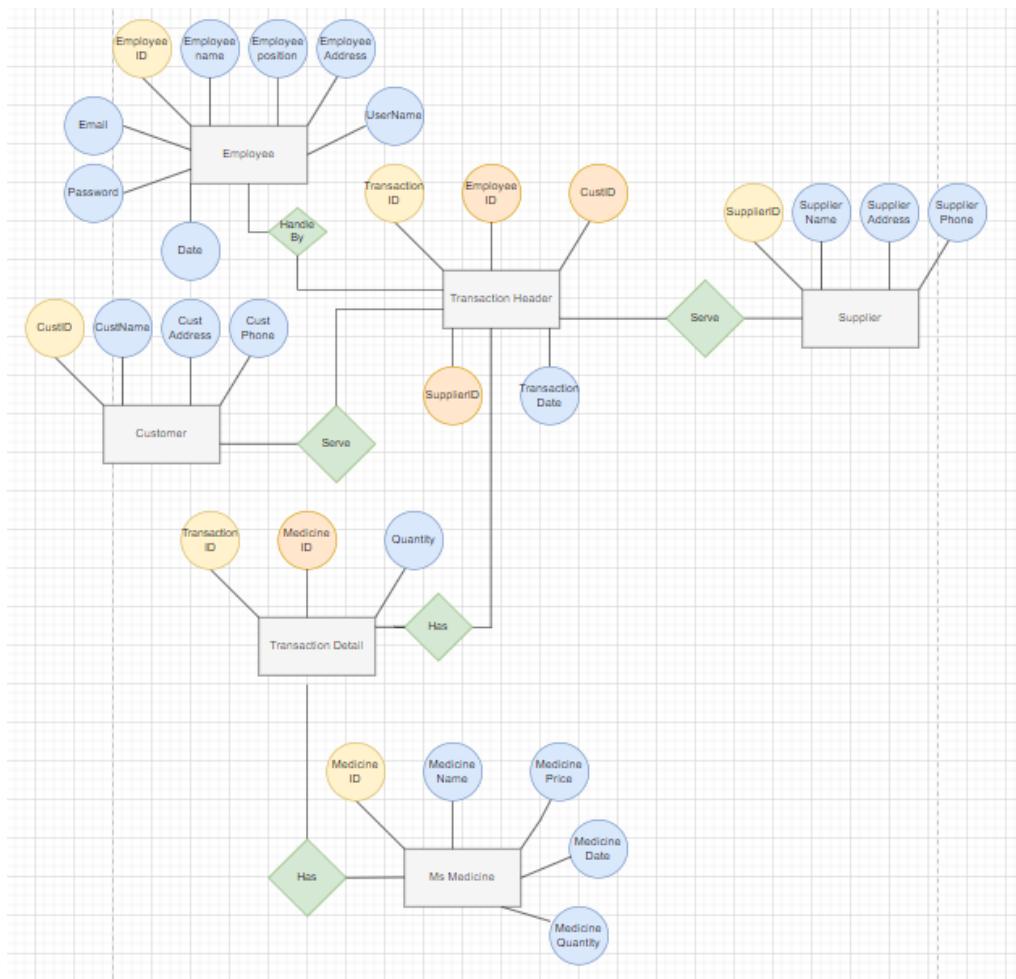
3.1.3. 2NF

53	2NF						
54	Customer						
55	CustID	CustName	CustAddress	CustPhone			
56	C0001	Willyam	Gading Serpong, Tangerang	0123-4567-8910			
57							
58	Medicine						
59	MedicineID	MedicineName	MedicinePrice	MedicineQty	MedicineDate		
60	M0001	Inhaler	IDR 50,000	100	22-Juli-2027		
61	M0002	Utrogestan	IDR 30,000	20	23-Agustus-2026		
62	M0003	Supralysin	IDR 50,000	20	15-Oktober-2027		
63	M0004	Tremenza	IDR 50,000	50	8-September-2026		
64	M0005	Favipiravir	IDR 60,000	10	17-November-2028		
65							
66	Employee						
67	EmployeeID	EmployeeName	EmployeeAddress	UserName	Password	Email	Date Join
68	EM0001	Farhan Korapat	BSD Alam Sutra, Tangerang	Louise	database_04	louise123@gmail.com	22-Apr-2022
69							
70	Supplier						
71	SupplierID	SupplierName	SupplierAddress	SupplierPhone			
72	SP0001	PT. Kimia Farma	couverland Barat Raya, Jakarta Utara	021-7759-4360			
73							
74	Transaction						
75	TransactionID	TransactionDate	Qty	Amount	Subtotal	Discount	Total
76	TR001	25-Apr-2022	200	IDR 9,000,000	IDR 9,700,000	IDR 0	IDR 9,700,000
77	TR001	25-Apr-2022	200	IDR 800,000	IDR 9,700,000	IDR 0	IDR 9,700,000
78	TR001	25-Apr-2022	200	IDR 1,000,000	IDR 9,700,000	IDR 0	IDR 9,700,000
79	TR001	25-Apr-2022	200	IDR 2,500,000	IDR 9,700,000	IDR 0	IDR 9,700,000
80	TR001	25-Apr-2022	200	IDR 800,000	IDR 9,700,000	IDR 0	IDR 9,700,000
81							

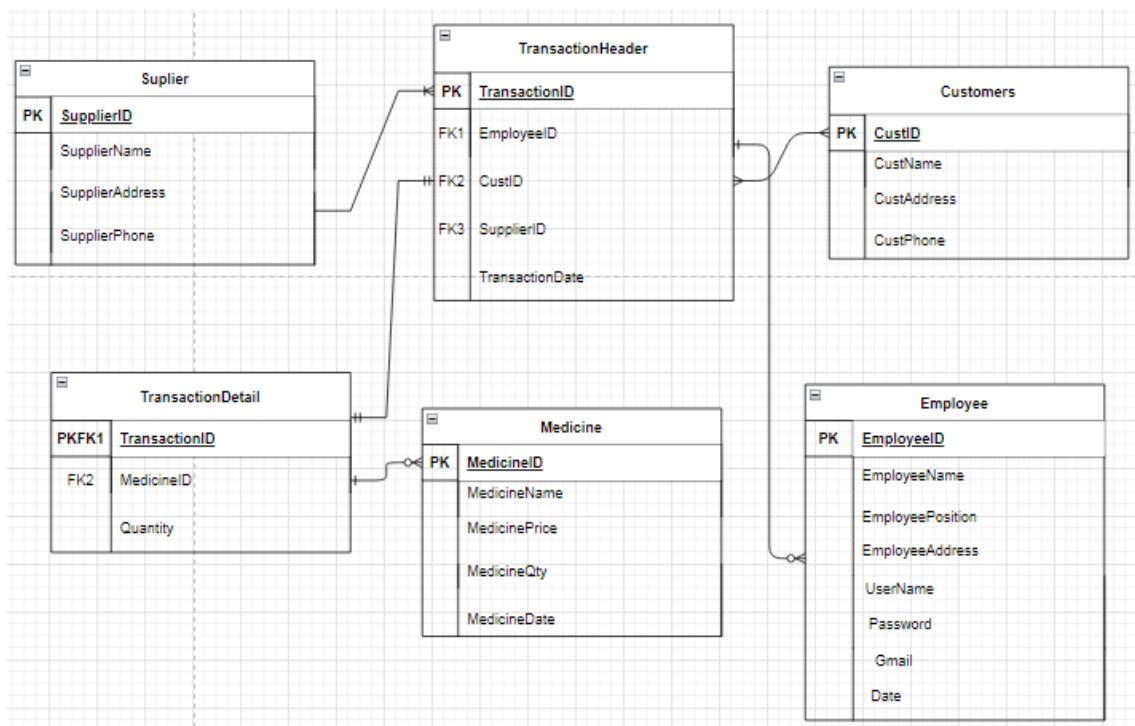
3.1.4. 3NF

82	3NF						
83	Transaction_Header						
84	TransactionID	CustID	EmployeeID	SupplierID	TransactionDate		
85	TR001	C0001	EM0001	SP0001	25-Apr-2022		
86							
87	Transaction_Detail						
88	TransactionID	MedicineID	Qty	Amount	Subtotal	Discount	Total
89	TR001	M0001	200	IDR 5,000,000	IDR 9,700,000	IDR 0	IDR 9,700,000
90	TR001	M0002	200	IDR 600,000	IDR 9,700,000	IDR 0	IDR 9,700,000
91	TR001	M0003	200	IDR 1,000,000	IDR 9,700,000	IDR 0	IDR 9,700,000
92	TR001	M0004	200	IDR 2,500,000	IDR 9,700,000	IDR 0	IDR 9,700,000
93	TR001	M0005	200	IDR 600,000	IDR 9,700,000	IDR 0	IDR 9,700,000
94							
95	Medicine						
96	MedicineID	MedicineName	Medicine Price	MedicineQty			
97	M0001	Inhaler	IDR 50,000	100			
98	M0002	Utrogestan	IDR 30,000	20			
99	M0003	Supralysin	IDR 50,000	20			
100	M0004	Tremenza	IDR 50,000	50			
101	M0005	Favipiravir	IDR 60,000	10			
102							
103	Customer						
104	CustID	CustName	CustAddress	CustPhone			
105	C0001	Willyam	Gading Serpong, Tangerang	0123-4567-8910			
106							
107	Employee						
108	EmployeeID	EmployeeName	UserName	Password	Email	Date Join	EmployeePosition
109	EM0001	Farhan Korapat	Louise	database_04	louise123@gmail.com	22-Apr-2022	Karyawan
110							BSD Alam Sutra, Tangerang
111	Supplier						
112	SupplierID	SupplierName	SupplierAddress	SupplierPhone			
113	SP0001	PT. Kimia Farma	couverland Barat Raya, Jakarta Utara	021-7759-4360			
114							

3.2. Entity Relationship Modeling



3.3. Enhanced Entity Relationship Diagram



Penjelasan:

1. UNF (*Unnormalized Form*)

Pada *unnormailized form*, terdapat data yang akan menjadi *header* pada setiap tabel transaksi yang pada umumnya muncul berulang kali untuk sebuah transaksi yang sama akan digabungkan menjadi satu *cell*, seperti id, password, email..

2. 1NF (*First Normal Form*)

Pada *First Normal Form*, terdapat penggabungan data yang terdapat pada *unnormailized form* menjadi satu *cell* yang akan dipecah setiap baris sehingga aturan *first normal form* dapat terpenuhi. Aturan *first normal form* tersebut berupa sebuah relasi dimana setiap irisan antara kolom dan baris telah berisikan hanya satu nilai. Dalam hal ini, terdapat *candidate key* berupa *primary key* pada tabel transaksi sebagai tabel *first normal form*, yaitu atribut id dan id_transaksi

3. 2NF (*Second Normal Form*)

Pada *second normal form*, proses normalisasi yang dilakukan telah menghasilkan dua buah tabel sebagai hasil pemecahan dari tabel *first normal form*, yaitu tabel dan tabel *Transaction* dengan tabel *Customer*, *Medicine*, *Employee*, dan *users*. Proses ini melibatkan pemisahan antara atribut yang bersifat *partially dependent* ataupun *fully dependent* terhadap suatu *candidate key* yang akan dilakukan sehingga menghasilkan tabel baru sesuai dengan tingkat ketergantungan pada setiap atribut, yaitu sebagai berikut:

- **Tabel *Customer*** yang tercipta oleh karena adanya atribut ***CustID***, ***CustName***, ***CustAddress***, ***CustPhone*** yang bersifat *partially dependent* terhadap atribut ***id***.
- **Tabel *Medicine*** yang tercipta oleh karena adanya atribut ***MedicineID***, ***MedicineName***, ***MedicinePrice***, ***MedicineDate***, ***MedicineQty*** yang bersifat *partially dependent* terhadap atribut ***id***.
- **Tabel *Employee*** yang tercipta oleh karena adanya atribut ***EmployeeID***, ***EmployeeName***, ***EmployeePosition***, ***EmployeeAddress***, ***Username***,

Password, Email, Date yang bersifat *partially dependent* terhadap atribut **id**.

- **Tabel Supplier** yang tercipta oleh karena adanya atribut **SupplierID, SupplierName, SupplierAddress, SupplierPhone** yang bersifat *partially dependent* terhadap atribut **id**.
- **Tabel Transaction** yang tercipta oleh karena adanya atribut **TransactionID, Qty, Amount, SubTotal, TransactionDate, Total** yang bersifat *partially dependent* terhadap atribut **id**.

4. 3NF (*Third Normal Form*)

Pada *third normal form*, terdapat *non-primary key* yang memiliki *transitive dependencies* terhadap *primary key* dalam suatu tabel dalam proses normalisasi pada *second normal form*. Oleh karena itu, dibutuhkan pemecahan secara terperinci menjadi sebuah tabel baru.

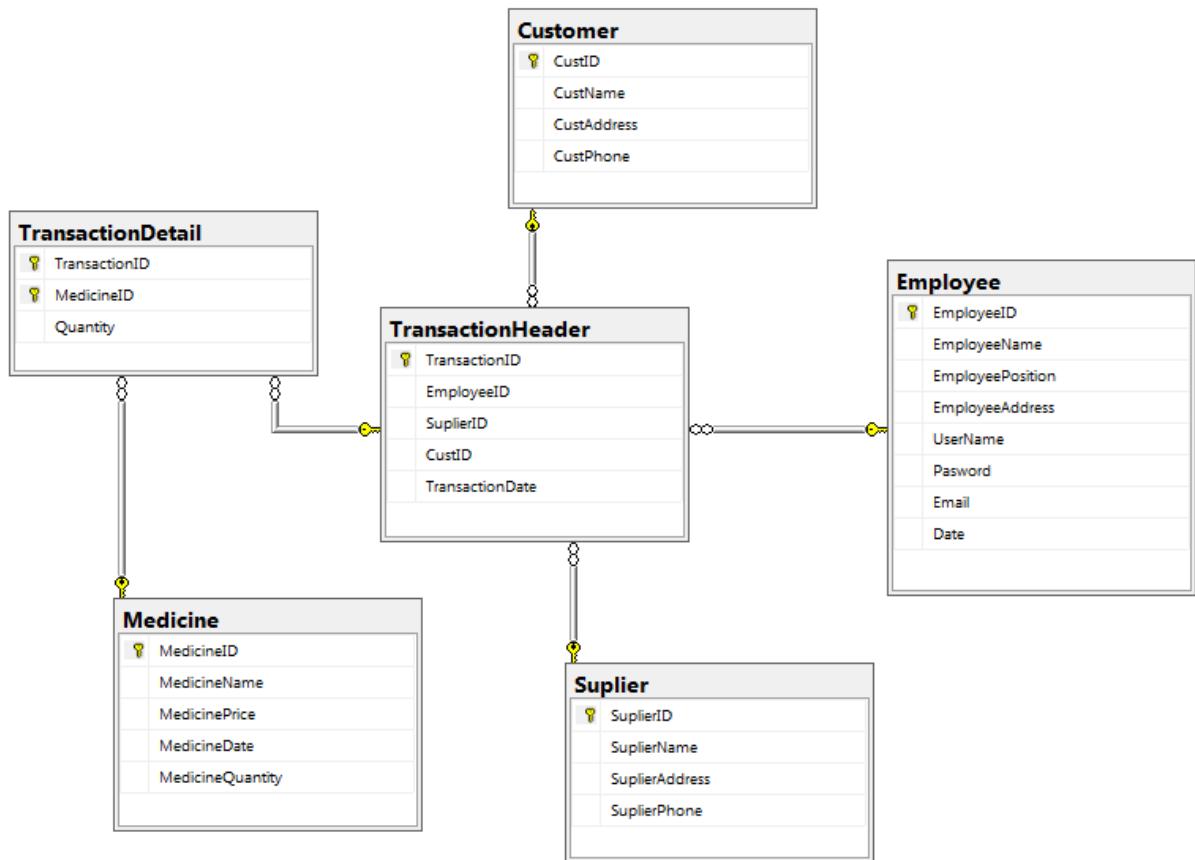
- Atribut **CustID, CustName, CustAddress, CustPhone** yang bersifat transitive dependent terhadap primary key **CustID** dalam tabel **Transaction_Header** sehingga hal ini menimbulkan terciptanya tabel baru, yaitu **tabel Customer**.
- Atribut **MedicineID, MedicineName, MedicinePrice, MedicineDate, MedicineQty** yang bersifat transitive dependent terhadap primary key **MedicineID** dalam tabel **Transaction_Detail** sehingga hal ini menimbulkan terciptanya tabel baru, yaitu **tabel Medicine**.
- Atribut **EmployeeID, EmployeeName, EmployeePosition, EmployeeAddress, Username, Password, Email, DateJoin** yang bersifat transitive dependent terhadap primary key **EmployeeID** dalam tabel **Transaction_Header** sehingga hal ini menimbulkan terciptanya tabel baru, yaitu **tabel Employee**.
- Atribut **SupplierID, SupplierName, SupplierAddress, SupplierPhone** yang bersifat transitive dependent terhadap primary key **EmployeeID** dalam tabel **Transaction_Header** sehingga hal ini menimbulkan terciptanya tabel baru, yaitu **tabel Supplier**.

- Atribut *TransactionID*, *MedicineID*, *Qty*, *Amount*, *Subtotal*, *Discount*, *Total* yang bersifat transitive dependent terhadap primary key *TransactionID* dalam tabel *Transaction_Header* sehingga hal ini menimbulkan terciptanya tabel baru, yaitu tabel *Transaction_Detail*.
- Atribut *TransactionID*, *CustID*, *EmployeeID*, *SupplierID*, dan *TransactionDate* pada tabel *Transaction_Header* yang dimana dari semua tabel yang ada merupakan *Foreign Key* kecuali *TransactionDate* dari setiap tabel yaitu *TransactionID* dari tabel *Transaction_Detail*, *CustID* dari tabel *Customer*, *EmployeeID* dari tabel *Employee* dan *SupplierID* dari tabel *Supplier*.

BAB IV

PENYELESAIAN MASALAH

4.1. Tabel Relasi



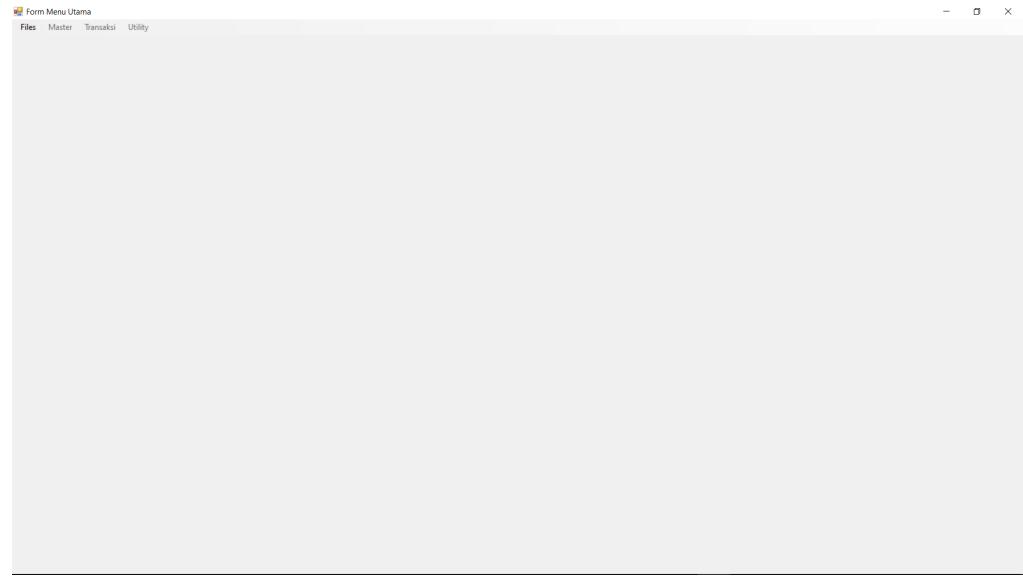
4.2. Penjelasan & Gambar User Interface Aplikasi & Kodingan

4.2.1. Mdi Form

Berikut ini merupakan tampilan awal saat aplikasi dijalankan. Pada Form ini terdapat 4 menu utama :

1. Main Menu File : Terdapat Login, Logout, dan Exit.
2. Main Menu Master : Terdapat Obat, Employee, Customer, Supplier.
3. Main Menu Transaksi : Terdapat Transaksi Kasir.
4. Main Menu Utility : Terdapat About.

Sebelum itu User harus melakukan login untuk mengakses menu Master, Transaksi, dan Utility. Jika belum melakukan Login maka Menu-Menu tersebut akan tidak dapat diakses.



Berikut merupakan codingan dalam membuat Form Menu Utama Untuk Code MenuTerkunci() agar bagian menu Master, Transaksi, dan Utility tidak dapat diakses.

```
public partial class FormMenuUtama : Form
{
    public static FormMenuUtama menu;
    MenuStrip mnstrip;

    void MenuTerkunci()
    {
        menuLogin.Enabled = true;
        menuLogout.Enabled = false;
        menuMaster.Enabled = false;
        menuTransaksi.Enabled = false;
        menuUtility.Enabled = false;
        menu = this;
    }

    FormLogin vformLogin;
    void vformLogin_formClosed(object sender, FormClosedEventArgs e)
    {
        vformLogin = null;
    }

    FormStokObat vformObat;
    void vformObat_formClosed(object sender, FormClosedEventArgs e)
    {
        vformObat = null;
    }

    FormEmployee vformEmployee;
    void vformEmployee_formClosed(object sender, FormClosedEventArgs e)
    {
        vformEmployee = null;
    }

    FormCustomer vformCustomer;
    void vformCustomer_formClosed(object sender, FormClosedEventArgs e)
    {
        vformCustomer = null;
    }
}
```

Code berikut dibuat agar Form tidak double saat membuka menu lain. Jadi Form tersebut jika ingin membuka Form lain harus di Close terlebih dahulu.

```
FormSupplier vformSupplier;
void vformSupplier_formClosed(object sender, FormClosedEventArgs e)
{
    vformSupplier = null;
}

FormTransaksi vformTransaksi;
void vformTransaksi_formClosed(object sender, FormClosedEventArgs e)
{
    vformTransaksi = null;
}

FormAbout vformAbout;
void vformAbout_formClosed(object sender, FormClosedEventArgs e)
{
    vformAbout = null;
}

public FormMenuUtama()
{
    InitializeComponent();
}

private void loginToolStripMenuItem_Click(object sender, EventArgs e)
{
    if (vformLogin == null)
    {
        vformLogin = new FormLogin();
        vformLogin.FormClosed += new FormClosedEventHandler(vformLogin_formClosed);
        vformLogin.ShowDialog();
    }
    else
    {
        vformLogin.Activate();
    }
}

private void FormMenuUtama_Load(object sender, EventArgs e)
{
    MenuTerkunci();
}

private void menuExit_Click(object sender, EventArgs e)
{
    Application.Exit();
}

private void menuLogout_Click(object sender, EventArgs e)
{
    MenuTerkunci();
}

private void obatToolStripMenuItem_Click(object sender, EventArgs e)
{
    if (vformObat == null)
    {
        vformObat = new FormStokObat();
        vformObat.FormClosed += new FormClosedEventHandler(vformObat_formClosed);
        vformObat.ShowDialog();
    }
    else
    {
        vformObat.Activate();
    }
}

private void employeeToolStripMenuItem_Click(object sender, EventArgs e)
{
    if (vformEmployee == null)
    {
        vformEmployee = new FormEmployee();
```

```

        vformEmployee.FormClosed += new FormClosedEventHandler(vformEmployee_formClosed);
        vformEmployee.ShowDialog();
    }
    else
    {
        vformEmployee.Activate();
    }
}

private void customerToolStripMenuItem_Click(object sender, EventArgs e)
{
    if (vformCustomer == null)
    {
        vformCustomer = new FormCustomer();
        vformCustomer.FormClosed += new FormClosedEventHandler(vformCustomer_formClosed);
        vformCustomer.ShowDialog();
    }
    else
    {
        vformCustomer.Activate();
    }
}

private void supplierToolStripMenuItem_Click(object sender, EventArgs e)
{
    if (vformSupplier == null)
    {
        vformSupplier = new FormSupplier();
        vformSupplier.FormClosed += new FormClosedEventHandler(vformSupplier_formClosed);
        vformSupplier.ShowDialog();
    }
    else
    {
        vformSupplier.Activate();
    }
}

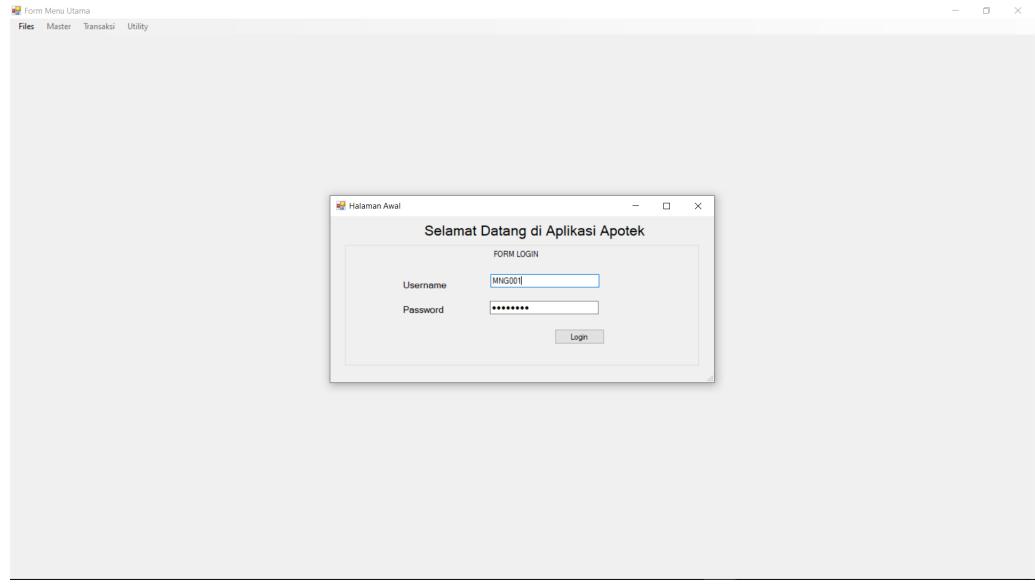
private void menuTransaksi_Click(object sender, EventArgs e)
{
    if (vformTransaksi == null)
    {
        vformTransaksi = new FormTransaksi();
        vformTransaksi.FormClosed += new FormClosedEventHandler(vformTransaksi_formClosed);
        vformTransaksi.ShowDialog();
    }
    else
    {
        vformTransaksi.Activate();
    }
}

private void aboutToolStripMenuItem_Click(object sender, EventArgs e)
{
    if (vformAbout == null)
    {
        vformAbout = new FormAbout();
        vformAbout.FormClosed += new FormClosedEventHandler(vformAbout_formClosed);
        vformAbout.ShowDialog();
    }
    else
    {
        vformAbout.Activate();
    }
}
}

```

4.2.2. Login

Kami membuat aplikasi kasir sederhana, dengan memakai data yang ada di dalam apotek kimia farma. Pada Halaman awal, kita harus Log In terlebih dahulu untuk memastikan apa yang mau anda lakukan. Sebelum anda login, akun telah dibuatkan terlebih dahulu oleh pihak berwenang.



Berikut merupakan tahap yang digunakan untuk membuat Form Login ini :

```
public partial class FormLogin : Form
{
    string koneksi = "Data Source=FERROX\\SQLEXPRESS;Initial Catalog=UAS_Visprog;Integrated Security=True";
    DataSet ds = new DataSet();
    SqlCommand cmd;
    SqlDataAdapter adpt;
    DataTable dt;
    string vquery = "";

    public FormLogin()
    {
        InitializeComponent();
    }

    private void button2_Click(object sender, EventArgs e)
    {
        FormStokObat fso = new FormStokObat();
        fso.Show();
    }

    DataClasses1DataContext db = new DataClasses1DataContext();

    private void button1_Click(object sender, EventArgs e)
    {
        DataClasses1DataContext db = new DataClasses1DataContext();
        var login = (from a in db.Employees
                    where a.UserName == txtUsername.Text
                    && a.Password == txtPassword.Text
                    select a);

        if (login.Any())
        {
            FormMenuUtama.menu.menuLogin.Enabled = false;
            FormMenuUtama.menu.menuLogout.Enabled = true;
            FormMenuUtama.menu.menuMaster.Enabled = true;
        }
    }
}
```

```

        FormMenuUtama.menu.menuTransaksi.Enabled = true;
        FormMenuUtama.menu.menuUtility.Enabled = true;

        MessageBox.Show("Login Data Berhasil");
        this.Close();
    }
}
else
{
    MessageBox.Show("Login Data Gagal");
}

}

private void button2_Click_1(object sender, EventArgs e)
{
    FormSupplier fs = new FormSupplier();
    fs.Show();
}

private void button2_Click_2(object sender, EventArgs e)
{
    MainMenu mm = new MainMenu();
    mm.Show();
}

private void FormLogin_Load(object sender, EventArgs e)
{
    txtUsername.Text = "MNG001";
    txtPassword.Text = "admin123";
}

private void groupBox1_Enter(object sender, EventArgs e)
{

}

private void label8_Click(object sender, EventArgs e)
{
}

private void label10_Click(object sender, EventArgs e)
{
}
}

```

Pada Form Login di atas, kita menggunakan *LINQ* untuk melakukan login untuk mengecek *username* dan *password* agar sesuai dari yang di input dengan yang ada di database. dimana username dan passwordnya cocok, nanti akan meng enable semua menu yang ada di MDI Form.

4.2.3. Medicine

Jika menekan button Medicine dari Form Pemilihan, berarti anda ingin menambah, update, dan menghapus data obat yang diinginkan. bottom tersebut berguna untuk mengecek obat - obatan ataupun semua data yang berada pada table Medicine. Berikut beberapa button yang digunakan :

1. Create : Jika data medicine belum pernah diinput.
2. Update : Jika ada data yang ingin diubah button Update bisa digunakan.
3. Delete : Menghapus data medicine yang tidak lagi dibutuhkan.
4. Cari : Digunakan untuk mencari data medicine berdasarkan id medicine.
5. Reset : Digunakan untuk merefresh tampilan database di GridView.
6. Pilih data : Digunakan untuk memilih data untuk di Update / Delete.
7. Cetak Report Tanggal : Menekan Button ini akan diarahkan ke List Obat dengan Crystal Report dengan tampilan data medicine berdasarkan tanggal.
8. Cetak Report : Menekan Button ini akan diarahkan ke List Obat dengan Crystal Report.

EmployeeID	EmployeeName	EmployeePosition	EmployeeAddress	UserName	Password	Email	Date
EMP001	Rifki	Manager	Bogor	MNG001	admin123	test@gmail.com	2001-12-12
EMP002	Rifki Dwiyara.	Employee	Rifki Dwiyara.	Ferox	admin123	test@gmail.com	07/06/2022

Berikut merupakan tahapan dalam pembuatan Form Insert Gudang.

```

public partial class FormStokObat : Form
{
    string koneksi = "Data Source=FERROX\\SQLEXPRESS;Initial Catalog=UAS_Vispog;Integrated Security=True";
    DataSet ds = new DataSet();
    SqlCommand cmd;
    SqlDataAdapter adpt;
    DataTable dt;
    string vquery = "";

    Koneksi konn = new Koneksi();
    private SqlDataAdapter da;
    private SqlDataReader rd;

    public FormStokObat()
    {
        InitializeComponent();
    }

    private void button2_Click(object sender, EventArgs e) {

        DataClasses1DataContext db = new DataClasses1DataContext();
        Medicine mdc = new Medicine
        {
            MedicineID = txtMedId.Text,
            MedicineName = txtMedNama.Text,
            MedicinePrice = (Convert.ToInt32(txtMedPrice.Text)),
            MedicineQty = (Convert.ToInt32(txtMedQuantity.Text)),
            MedicineDate = this.dateTimePicker1.Text,
        };
        db.Medicines.InsertOnSubmit(mdc);
        try
        {
            db.SubmitChanges();
            MessageBox.Show("Input data berhasil");

            txtMedId.Clear();
            txtMedNama.Clear();
            txtMedPrice.Clear();
            txtMedQuantity.Clear();
            dateTimePicker1.CustomFormat="";
            showData();
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.ToString());
        }
    }

    public void showData() {
        adpt = new SqlDataAdapter("select * from Medicine", koneksi);
        dt = new DataTable();
        adpt.Fill(dt);
        dataGridView1.DataSource = dt;
    }

    private void groupBox1_Enter(object sender, EventArgs e)
    {
    }

    private void dataGridView1_CellContentClick(object sender, DataGridViewCellEventArgs e)
    {
    }

    private void txtMedId_Leave(object sender, EventArgs e)
    {
    }
}

```

```

        }

        private void FormStokObat_Load(object sender, EventArgs e)
        {
            showData();
            NoOtomatis();
            dataGridView1.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill;
        }

        void NoOtomatis()
        {
            long hitung;
            string urutan;
            SqlDataReader rd;
            SqlConnection conn = konn.GetConn();
            conn.Open();
            cmd = new SqlCommand("SELECT MedicineID FROM Medicine WHERE MedicineID IN(SELECT MAX (MedicineID) FROM Medicine) ORDER BY MedicineID DESC", conn);
            rd = cmd.ExecuteReader();
            rd.Read();
            if (rd.HasRows)
            {
                hitung = Convert.ToInt64(rd[0].ToString().Substring(rd["MedicineID"].ToString().Length - 3, 3)) + 1;
                string kodeUrutan = "000" + hitung;
                urutan = "MDI" + kodeUrutan.Substring(kodeUrutan.Length - 3, 3);
            }
            else
            {
                urutan = "MDI001";
            }
            rd.Close();
            txtMedId.Text = urutan;
            conn.Close();
        }

        private void button3_Click(object sender, EventArgs e)
        {

DataClasses1DataContext db = new DataClasses1DataContext();
var ubah = (from a in db.Medicines
           where a.MedicineID == txtMedId.Text
           select a).Single();
ubah.MedicineName = txtMedNama.Text;
ubah.MedicinePrice = (Convert.ToInt32(txtMedPrice.Text));
ubah.MedicineQty = (Convert.ToInt32(txtMedQuantity.Text));
try
{
    db.SubmitChanges();
    MessageBox.Show("Update Data Berhasil");

    txtMedId.Clear();
    txtMedNama.Clear();
    txtMedPrice.Clear();
    txtMedQuantity.Clear();
    dateTimePicker1.CustomFormat = "";
    showData();

}
catch (Exception ex)
{
    MessageBox.Show(ex.ToString());
}
}

private void button4_Click(object sender, EventArgs e)
{
    DataClasses1DataContext db = new DataClasses1DataContext();
    var delete = (from a in db.Medicines
                  where a.MedicineID == txtMedId.Text
                  select a).Single();
    db.Medicines.DeleteOnSubmit(delete);
    try
    {
        db.SubmitChanges();
    }
}

```

```
        MessageBox.Show("Delete Data Berhasil");

        txtMedId.Clear();
        txtMedNama.Clear();
        txtMedPrice.Clear();
        txtMedQuantity.Clear();
        dateTimePicker1.CustomFormat = "";
        showData();
    }
}
catch (Exception ex)
{
    MessageBox.Show(ex.ToString());
}
}

private void button5_Click(object sender, EventArgs e)
{
    ReportViewerMedicine rvm = new ReportViewerMedicine();
    rvm.Show();
}

private void button6_Click(object sender, EventArgs e)
{
    txtMedId.Text = this.dataGridView1.CurrentRow.Cells[0].Value.ToString();
    txtMedNama.Text = this.dataGridView1.CurrentRow.Cells[1].Value.ToString();
    txtMedPrice.Text = this.dataGridView1.CurrentRow.Cells[2].Value.ToString();
    txtMedQuantity.Text = this.dataGridView1.CurrentRow.Cells[3].Value.ToString();

}

private void button1_Click(object sender, EventArgs e)
{
    this.Close();
}

private void dataGridView1_CellContentClick_1(object sender, DataGridViewCellEventArgs e)
{
}

private void txtMedId_TextChanged(object sender, EventArgs e)
{
}

private void button7_Click(object sender, EventArgs e)
{
    ReportViewerMedicine2 rvm2 = new ReportViewerMedicine2();
    rvm2.Show();
}

private void button8_Click(object sender, EventArgs e)
{
    adpt = new SqlDataAdapter("select * from Medicine WHERE MedicineID = '" + txtSearch.Text + "'", koneksi);
    dt = new DataTable();
    adpt.Fill(dt);
    dataGridView1.DataSource = dt;
}

private void button9_Click(object sender, EventArgs e)
{
    showData();
}

private void lblid_Click(object sender, EventArgs e)
{
}

private void txtSearch_TextChanged(object sender, EventArgs e)
{
}

}
```

Pada Code diatas kami menggunakan LINQ untuk proses create, update dan delete. Dimana ada sebuah file LINQ yaitu DataClases1.dbml sebagai tempat data tersimpan. Setiap code akan digunakan menyesuaikan dengan proses

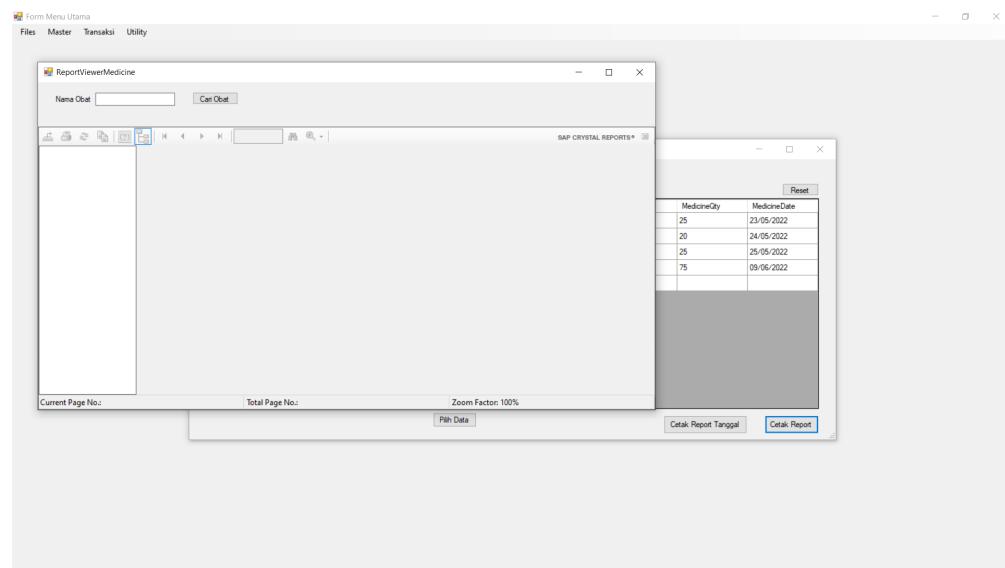
yang ingin dilakukan seperti InsertOnSubmit untuk melakukan create, membuat query ubah untuk melakukan update, dan DeleteOnSubmit untuk melakukan Delete pada form Medicine.

Lalu pada button 6, yaitu button pilih, kami menggunakan `this.dataGridView1.CurrentRow.Cells[].Value.ToString();` untuk mengambil data dari gridview dan dipindahkan kedalam textbox agar mempermudah pengguna untuk mengupdate dan delete data..

Untuk menampilkan data, kami menggunakan SqlDataAdapter untuk menuliskan query, yang nantinya akan disimpan kedalam dataTable, dan akan digunakan sebagai value GridView.

Lalu kami juga membuat program auto increment untuk setiap ID yang ada di semua form. Dimana kami membuatnya dengan mengambil id yang telah ada sebelumnya di database. setelah itu id nya akan dibagi menjadi 2 bagian yaitu memisahkan antara alfabet dengan numeric, dimana numeric yang ada di database akan di tambah 1. sehingga terjadi auto increment dengan program c#. dan sisanya adalah code untuk menampilkan form lain.

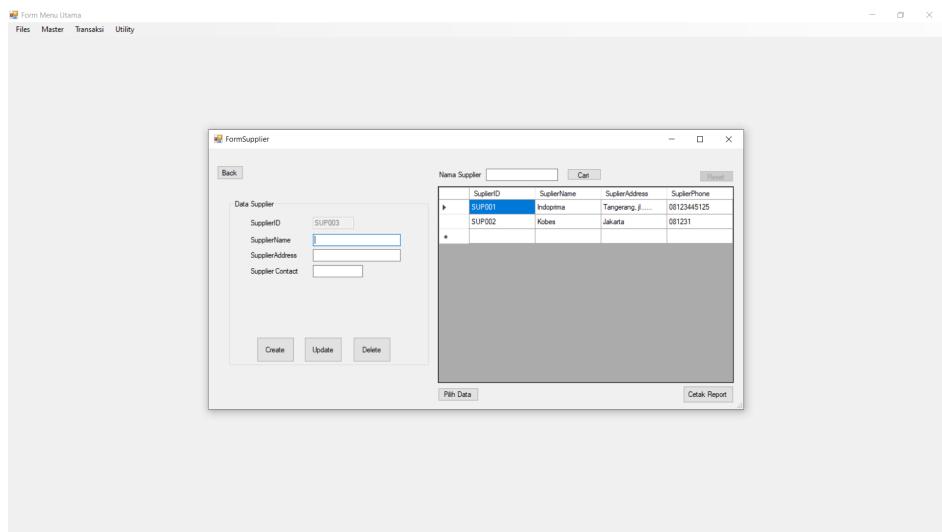
Jika menekan tombol Cetak Report Tanggal / Cetak Report maka akan diarahkan ke List Obat dengan Crystal Report



4.2.4. Supplier

Jika menekan button Supplier, berarti anda bisa melakukan beberapa hal seperti yang ditunjukkan pada **FormSupplier**. Form ini berfungsi untuk mengecek data supplier yang sebelumnya pernah mengirim barang. yang nantinya, database tersebut akan ditampilkan di **DataGridView**. Berikut beberapa button yang digunakan :

1. Create : Jika data Supplier belum pernah diinput.
2. Update : Jika ada data yang ingin diubah button Update bisa digunakan.
3. Delete : Menghapus data Supplier yang tidak lagi dibutuhkan.
4. Cari : Digunakan untuk mencari data Supplier berdasarkan nama Supplier.
5. Reset : Digunakan untuk merefresh tampilan database di GridView.
6. Pilih data : Digunakan untuk memilih data untuk di Update / Delete.
7. Cetak Report : Menekan Button ini akan diarahkan ke List Supplier dengan Crystal Report.



Berikut merupakan kodingan dalam pembuatan **FormSupplier**.

```

public partial class FormSupplier : Form
{
    string koneksi = "Data Source=FERROX\\SQLEXPRESS;Initial Catalog=UAS_Visprog;Integrated Security=True";
    DataSet ds = new DataSet();
    SqlCommand cmd;
    SqlDataAdapter adpt;
    DataTable dt;
    string vquery = "";

    Koneksi konn = new Koneksi();
    private SqlDataAdapter da;
    private SqlDataReader rd;
    public FormSupplier()
    {
        InitializeComponent();
    }

    private void button2_Click(object sender, EventArgs e)
    {
        DataClasses1DataContext db = new DataClasses1DataContext();
        Suplier spp = new Suplier
        {
            SuplierID = txtSuppId.Text,
            SuplierName = txtSuppName.Text,
            SuplierAddress = txtSuppAdd.Text,
            SuplierPhone = txtSuppContact.Text,
        };
        db.Supliers.InsertOnSubmit(spp);
        try
        {
            db.SubmitChanges();
            MessageBox.Show("Input data berhasil");
            txtSuppId.Clear();

            txtSuppName.Clear();
            txtSuppAdd.Clear();
            txtSuppContact.Clear();
            showData();
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.ToString());
        }
    }

    void NoOtomatis()
    {
        long hitung;
        string urutan;
        SqlDataReader rd;
        SqlConnection conn = konn.GetConn();
        conn.Open();
        cmd = new SqlCommand("SELECT SuplierID FROM Suplier WHERE SuplierID IN(SELECT MAX (SuplierID) FROM Suplier) ORDER BY SuplierID DESC", conn);
        rd = cmd.ExecuteReader();
        rd.Read();
        if (rd.HasRows)
        {
            hitung = Convert.ToInt64(rd[0].ToString().Substring(rd["SuplierID"].ToString().Length - 3, 3)) + 1;
            string kodeUrutan = "000" + hitung;
            urutan = "SUP" + kodeUrutan.Substring(kodeUrutan.Length - 3, 3);
        }
        else
        {
            urutan = "SUP001";
        }
        rd.Close();
        txtSuppId.Text = urutan;
        conn.Close();
    }
}

```

```
public void showData()
{
    adapt = new SqlDataAdapter("select * from Suplier", koneksi);
    dt = new DataTable();
    adapt.Fill(dt);
    dataGridView1.DataSource = dt;
}

private void button3_Click(object sender, EventArgs e)
{
    DataClasses1DataContext db = new DataClasses1DataContext();
    var ubah = (from a in db.Supliers
                where a.SuplierID == txtSuppId.Text
                select a).Single();
    ubah.SuplierName = txtSuppName.Text;
    ubah.SuplierAddress = txtSuppAdd.Text;
    ubah.SuplierPhone = txtSuppContact.Text;
    try
    {
        db.SubmitChanges();
        MessageBox.Show("Update Data Berhasil");

        txtSuppId.Clear();
        txtSuppName.Clear();
        txtSuppAdd.Clear();
        txtSuppContact.Clear();
        showData();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.ToString());
    }
}
```

```

private void button4_Click(object sender, EventArgs e)
{
    DataClasses1DataContext db = new DataClasses1DataContext();
    var delete = (from a in db.Supliers
                  where a.SupperID == txtSuppId.Text
                  select a).Single();
    db.Supliers.DeleteOnSubmit(delete);
    try
    {
        db.SubmitChanges();
        MessageBox.Show("Delete Data Berhasil");
        txtSuppId.Clear();
        txtSuppName.Clear();
        txtSuppAdd.Clear();
        txtSuppContact.Clear();
        showData();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.ToString());
    }
}

private void button1_Click(object sender, EventArgs e)
{
    this.Close();
}

private void button5_Click(object sender, EventArgs e)
{
    ReportViewerSupplier rvs = new ReportViewerSupplier();
    rvs.Show();
}

private void FormSupplier_Load(object sender, EventArgs e)
{
    showData();
    Notomatics();
    dataGridView1.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill;
}

private void groupBox1_Enter(object sender, EventArgs e)
{
}

private void button6_Click(object sender, EventArgs e)
{
    txtSuppId.Text = this.dataGridView1.CurrentRow.Cells[0].Value.ToString();
    txtSuppName.Text = this.dataGridView1.CurrentRow.Cells[1].Value.ToString();
    txtSuppAdd.Text = this.dataGridView1.CurrentRow.Cells[2].Value.ToString();
    txtSuppContact.Text = this.dataGridView1.CurrentRow.Cells[3].Value.ToString();
}

private void button8_Click(object sender, EventArgs e)
{
    adpt = new SqlDataAdapter("select * from Suplier WHERE SuplierName = '" + txtSearch.Text + "'", koneksi);
    dt = new DataTable();
    adpt.Fill(dt);
    dataGridView1.DataSource = dt;
}

private void txtSearch_TextChanged(object sender, EventArgs e)
{
}

private void button7_Click(object sender, EventArgs e)
{
    showData();
}
}

```

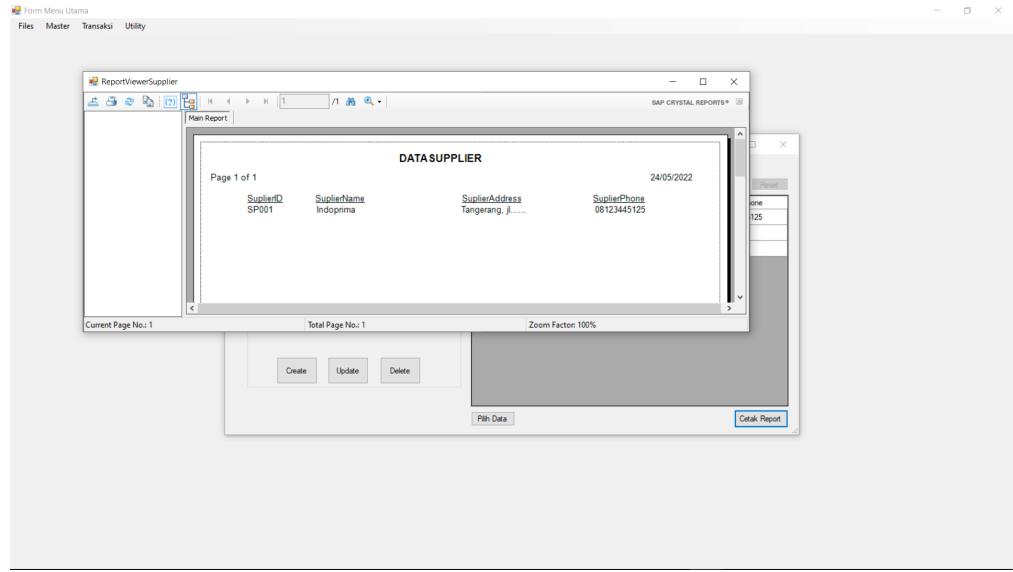
Sama seperti form obat, Pada Code diatas kami menggunakan LINQ untuk proses create, update dan delete. Dimana ada sebuah file LINQ yaitu DataClases1.dbml sebagai tempat data tersimpan. Setiap code akan digunakan menyesuaikan dengan proses yang ingin dilakukan seperti InsertOnSubmit untuk melakukan create, membuat query ubah untuk melakukan update, dan DeleteOnSubmit untuk melakukan Delete pada form Medicine.

Lalu pada button 6, yaitu button pilih, kami menggunakan `this.dataGridView1.CurrentRow.Cells[].Value.ToString();` untuk mengambil data dari gridview dan dipindahkan kedalam textbox agar mempermudah pengguna untuk mengupdate dan delete data..

Untuk menampilkan data, kami menggunakan SqlDataAdapter untuk menuliskan query, yang nantinya akan disimpan kedalam dataTable, dan akan digunakan sebagai value GridView.

Lalu kami juga membuat program auto increment untuk setiap ID yang ada di semua form. Dimana kami membuatnya dengan mengambil id yang telah ada sebelumnya di database. setelah itu id nya akan dibagi menjadi 2 bagian yaitu memisahkan antara alfabet dengan numeric, dimana numeric yang ada di database akan di tambah 1. sehingga terjadi auto increment dengan program c#. dan sisanya adalah code untuk menampilkan form lain.

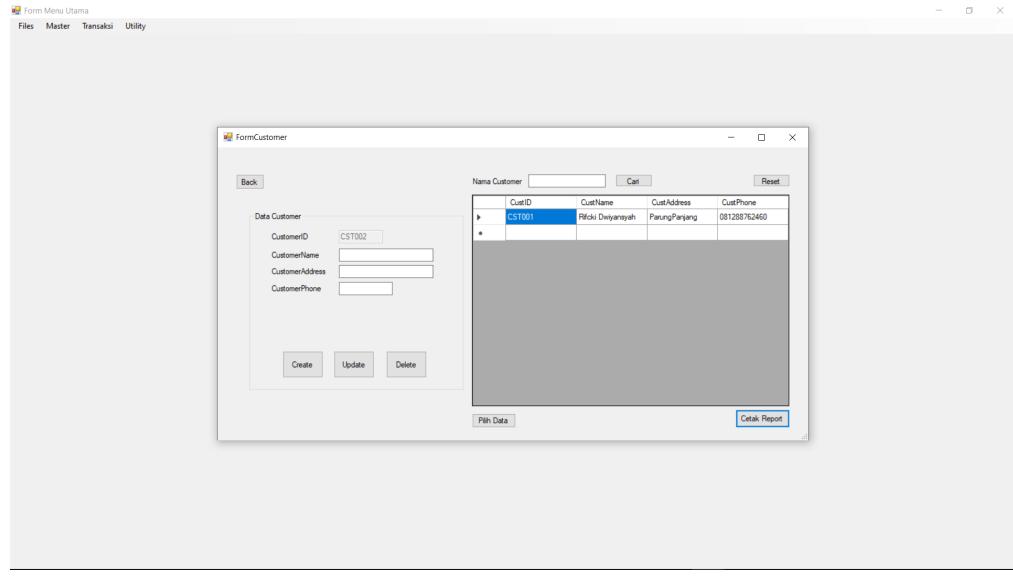
Jika menekan tombol report maka akan diarahkan ke List Supplier dengan Crystal Report dibawah ini.



4.2.5. Customer

Jika menekan button Customer dari Form Pemilihan, berarti kita bisa menambahkan data yang telah customer pesan, yang nantinya (dalam proses penggerjaan). Terdapat 3 button yang dapat dipakai yaitu :

1. Create : Jika data Customer belum pernah diinput.
2. Update : Jika ada data yang ingin diubah button Update bisa digunakan.
3. Delete : Menghapus data Customer yang tidak lagi dibutuhkan.
4. Cari : Digunakan untuk mencari data Customer berdasarkan nama Customer .
5. Reset : Digunakan untuk merefresh tampilan database di GridView.
6. Pilih data : Digunakan untuk memilih data untuk di Update / Delete.
7. Cetak Report : Menekan Button ini akan diarahkan ke List Customer dengan Crystal Report.



Berikut merupakan kodingan dalam pembuatan **FormCostumer**:

```

public partial class FormCustomer : Form
{
    string koneksi = "Data Source=FERROX\\SQLEXPRESS;Initial Catalog=UAS_Visprog;Integrated Security=True";
    DataSet ds = new DataSet();
    SqlCommand cmd;
    SqlDataAdapter adpt;
    DataTable dt;
    string vquery = "";

    Koneksi konn = new Koneksi();
    private SqlDataAdapter da;
    private SqlDataReader rd;

    public FormCustomer()
    {
        InitializeComponent();
    }

    void NoOtomatics()
    {
        long hitung;
        string urutan;
        SqlDataReader rd;
        SqlConnection conn = konn.GetConn();
        conn.Open();
        cmd = new SqlCommand("SELECT CustID FROM Customer WHERE CustID IN(SELECT MAX (CustID) FROM Customer) ORDER BY CustID DESC", conn);
        rd = cmd.ExecuteReader();
        rd.Read();
        if (rd.HasRows)
        {
            hitung = Convert.ToInt64(rd[0].ToString().Substring(rd["CustID"].ToString().Length - 3, 3)) + 1;
            string kodeUrutan = "000" + hitung;
            urutan = "CST" + kodeUrutan.Substring(kodeUrutan.Length - 3, 3);
        }
        else
        {
    
```

```
        urutan = "CST001";
    }
    rd.Close();
    txtCustID.Text = urutan;
    conn.Close();
}
public void showData()
{
    adpt = new SqlDataAdapter("select * from Customer", koneksi);
    dt = new DataTable();
    adpt.Fill(dt);
    dataGridView1.DataSource = dt;
}

private void button2_Click(object sender, EventArgs e)
{
    DataClasses1DataContext db = new DataClasses1DataContext();
    Customer cs = new Customer
    {
        CustID = txtCustID.Text,
        CustName = txtCustName.Text,
        CustAddress = txtCustAddress.Text,
        CustPhone = txtCustPhone.Text,
    };
    db.Customers.InsertOnSubmit(cs);
    try
    {
        db.SubmitChanges();
        MessageBox.Show("Input data berhasil");

        txtCustID.Clear();
        txtCustName.Clear();
        txtCustAddress.Clear();
        txtCustPhone.Clear();
        showData();
    }
}
```

```

        }

        catch (Exception ex)
        {
            MessageBox.Show(ex.ToString());
        }
    }

    private void button3_Click(object sender, EventArgs e)
{
    DataClasses1DataContext db = new DataClasses1DataContext();
    var ubah = (from a in db.Customers
                where a.CustID == txtCustID.Text
                select a).Single();
    ubah.CustName = txtCustName.Text;
    ubah.CustAddress = txtCustName.Text;
    ubah.CustPhone = txtCustPhone.Text;
    try
    {
        db.SubmitChanges();
        MessageBox.Show("Update Data Berhasil");
    }

    txtCustID.Clear();
    txtCustName.Clear();
    txtCustAddress.Clear();
    txtCustPhone.Clear();
    showData();
}
catch (Exception ex)
{
    MessageBox.Show(ex.ToString());
}
}

private void button4_Click(object sender, EventArgs e)
{
    DataClasses1DataContext db = new DataClasses1DataContext();
    var delete = (from a in db.Customers
                  where a.CustID == txtCustID.Text
                  select a).Single();
    db.Customers.DeleteOnSubmit(delete);
    try
    {
        db.SubmitChanges();
        MessageBox.Show("Delete Data Berhasil");

        txtCustID.Clear();
        txtCustName.Clear();
        txtCustAddress.Clear();
        txtCustPhone.Clear();
        showData();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.ToString());
    }
}

private void dataGridView1_CellContentClick(object sender, DataGridViewCellEventArgs e)
{
}

private void FormCustomer_Load(object sender, EventArgs e)
{
    showData();
    NoOtomatis();
    dataGridView1.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill;
}

```

```

private void button1_Click(object sender, EventArgs e)
{
    this.Close();
}

private void button5_Click(object sender, EventArgs e)
{
    ReportViewerCustomer rvc = new ReportViewerCustomer();
    rvc.Show();
}

private void button9_Click(object sender, EventArgs e)
{
    showData();
}

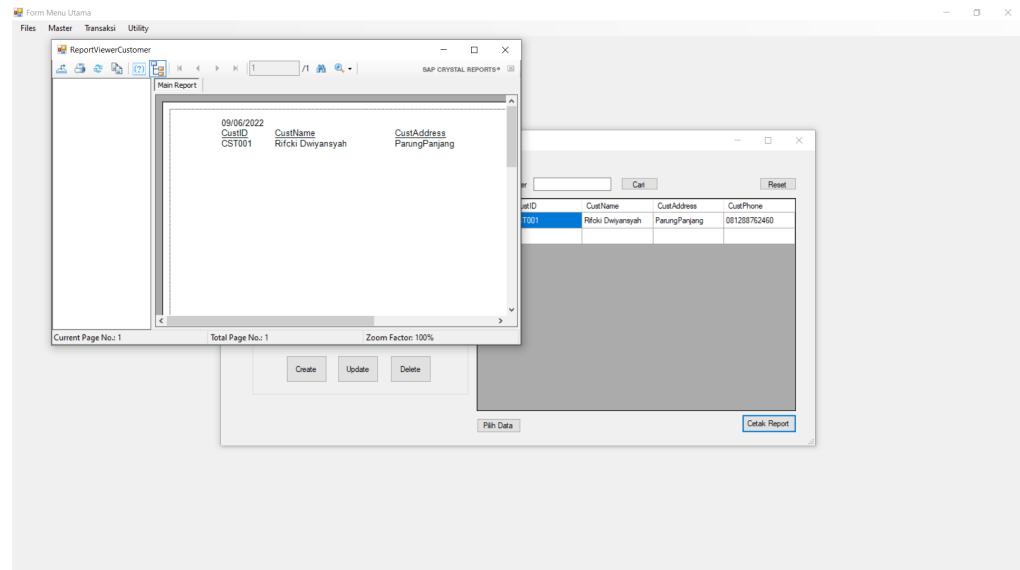
private void button8_Click(object sender, EventArgs e)
{
    adpt = new SqlDataAdapter("select * from Customer WHERE CustomerName = '" + txtSearch.Text + "'", koneksi);
    dt = new DataTable();
    adpt.Fill(dt);
    dataGridView1.DataSource = dt;
}

private void button6_Click(object sender, EventArgs e)
{
    txtCustID.Text = this.dataGridView1.CurrentRow.Cells[0].Value.ToString();
    txtCustName.Text = this.dataGridView1.CurrentRow.Cells[1].Value.ToString();
    txtCustAddress.Text = this.dataGridView1.CurrentRow.Cells[2].Value.ToString();
    txtCustPhone.Text = this.dataGridView1.CurrentRow.Cells[3].Value.ToString();
}

```

Untuk code yang kami gunakan, ini sama dengan form yang ada di Form Obat, dan Form Supplier.

Jika menekan tombol report maka akan diarahkan ke List Customer dengan Crystal Report dibawah ini.



Berikut merupakan tahapan yang dilakukan dalam pembuatan **Form Customer**:

4.2.6. Employee

Dalam form Employee ini, dapat melihat data pekerja yang ada. Pada form Employee ini juga terdapat pengaturan tanggal yang berguna kapan pekerja tersebut masuk dan keluar. Berikut adalah beberapa button yang dapat membantu input data :

1. Create : Jika data Employee belum pernah diinput.
2. Update : Jika ada data yang ingin diubah button Update bisa digunakan.
3. Delete : Menghapus data Employee yang tidak lagi dibutuhkan.
4. Cari : Digunakan untuk mencari data Employee berdasarkan nama Employee.
5. Reset : Digunakan untuk merefresh tampilan database di GridView.
6. Pilih data : Digunakan untuk memilih data untuk di Update / Delete.
7. Cetak Report : Menekan Button ini akan diarahkan ke List Employee dengan Crystal Report.

The screenshot shows the 'FormEmployee' application window. At the top, there's a menu bar with 'Form Menu Utama', 'Files', 'Master', 'Transaksi', and 'Utility'. Below the menu is a search bar labeled 'Nama Employee' with a 'Cari' button and a 'Reset' button. On the left, a panel titled 'Data Employee' contains fields for 'EmployeeID' (EMP003), 'EmployeeName' (Rikki), 'EmployeePosition' (Manager), 'EmployeeAddress' (Bogor), 'Username' (MNG001), 'Password' (admin123), 'Email' (test@gmail.com), and 'Tanggal Masuk' (09/06/2022). Below these fields are buttons for 'Create', 'Update', and 'Delete'. In the center, a 'GridView' displays a list of employees with columns: EmployeeID, EmployeeName, EmployeePosition, EmployeeAddress, UserName, Password, Email, and Date. The first two rows are visible: EMP001 (Rikki, Manager, Bogor, MNG001, admin123, test@gmail.com, 2001-12-12) and EMP002 (Rikki Dwiyana, Employee, Rikki Dwiyana, ferox, admin123, test@gmail.com, 07/06/2022). A 'Pilih Data' button is located at the bottom of the grid. At the very bottom of the window are 'Cetak Full' and 'Cetak Report' buttons.

Berikut merupakan tahapan yang dilakukan dalam pembuatan **Form Employee**:

```

public partial class FormEmployee : Form
{
    string koneksi = "Data Source=FERROX\\SQLEXPRESS;Initial Catalog=UAS_Visprog;Integrated Security=True";
    DataSet ds = new DataSet();
    SqlCommand cmd;
    SqlDataAdapter adpt;
    DataTable dt;
    string vquery = "";
    Koneksi konn = new Koneksi();
    private SqlDataAdapter da;
    private SqlDataReader rd;
    public FormEmployee()
    {
        InitializeComponent();
    }

    void NoOtomatis()
    {
        long hitung;
        string urutan;
        SqlDataReader rd;
        SqlConnection conn = konn.GetConn();
        conn.Open();
        cmd = new SqlCommand("SELECT EmployeeID FROM Employee WHERE EmployeeID IN(SELECT MAX (EmployeeID) FROM Employee) ORDER BY EmployeeID DESC", conn);
        rd = cmd.ExecuteReader();
        rd.Read();
        if (rd.HasRows)
        {
            hitung = Convert.ToInt64(rd[0].ToString().Substring(rd["EmployeeID"].ToString().Length - 3, 3)) + 1;
            string kodeUrutan = "000" + hitung;
            urutan = "EMP" + kodeUrutan.Substring(kodeUrutan.Length - 3, 3);
        }
        else
        {
            urutan = "EMP001";
        }
        rd.Close();
    }

    txtEmpID.Text = urutan;
    conn.Close();
}

public void showData()
{
    adpt = new SqlDataAdapter("select * from Employee", koneksi);
    dt = new DataTable();
    adpt.Fill(dt);
    dataGridView1.DataSource = dt;
}

private void button2_Click(object sender, EventArgs e)
{
    DataClasses1DataContext db = new DataClasses1DataContext();
    Employee emp = new Employee
    {
        EmployeeID = txtEmpID.Text,
        EmployeeName = txtEmpName.Text,
        EmployeeAddress = txtEmpName.Text,
        EmployeePosition = txtEmpPos.Text,
        UserName = txtUsername.Text,
        Password = txtPassword.Text,
        Email = txtEmail.Text,
        Date = this.dateTimePicker1.Text,
    };
    db.Employees.InsertOnSubmit(emp);
    try
    {
        db.SubmitChanges();
        MessageBox.Show("Input data berhasil");
        txtEmpID.Clear();
        txtEmpName.Clear();
        txtEmpPos.Clear();
        txtEmpAdd.Clear();
    }
}

```

```

        txtUsername.Clear();
        txtPassword.Clear();
        txtEmail.Clear();
        dateTimePicker1.CustomFormat = "";
        showData();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.ToString());
    }
}

private void button3_Click(object sender, EventArgs e)
{
    DataClasses1DataContext db = new DataClasses1DataContext();
    var ubah = (from a in db.Employees
                where a.EmployeeID == txtEmpID.Text
                select a).Single();
    ubah.EmployeeName = txtEmpID.Text;
    ubah.EmployeeAddress = txtEmpAdd.Text;
    ubah.EmployeePosition = txtEmpPos.Text;
    ubah.UserName = txtUsername.Text;
    ubah.Pasword = txtPassword.Text;
    ubah.Email = txtEmail.Text;
    try
    {
        db.SubmitChanges();
        MessageBox.Show("Update Data Berhasil");

        txtEmpID.Clear();
        txtEmpName.Clear();
        txtEmpPos.Clear();
        txtEmpAdd.Clear();
        txtUsername.Clear();
        txtPassword.Clear();
        txtEmail.Clear();

        dateTimePicker1.CustomFormat = "";
        showData();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.ToString());
    }
}

private void FormEmployee_Load(object sender, EventArgs e)
{
    showData();
    NoOtomatic();
    dataGridView1.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill;
}

private void dateTimePicker1_ValueChanged(object sender, EventArgs e)
{
}

private void label5_Click(object sender, EventArgs e)
{
}

private void button4_Click(object sender, EventArgs e)
{
    DataClasses1DataContext db = new DataClasses1DataContext();
    var delete = (from a in db.Employees
                 where a.EmployeeID == txtEmpID.Text
                 select a).Single();
    db.Employees.DeleteOnSubmit(delete);
    try
    {
}

```

```

        db.SubmitChanges();
        MessageBox.Show("Delete Data Berhasil");

        txtEmpID.Clear();
        txtEmpName.Clear();
        txtEmpPos.Clear();
        txtEmpAdd.Clear();
        txtUsername.Clear();
        txtPassword.Clear();
        txtEmail.Clear();
        dateTimePicker1.CustomFormat = "";
        showData();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.ToString());
    }
}

private void button1_Click(object sender, EventArgs e)
{
    this.Close();
}

private void button5_Click(object sender, EventArgs e)
{
    ReportViewEmployee fve = new ReportViewEmployee();
    fve.Show();
}

private void button6_Click(object sender, EventArgs e)
{
    txtEmpID.Text = this.dataGridView1.CurrentRow.Cells[0].Value.ToString();
    txtEmpName.Text = this.dataGridView1.CurrentRow.Cells[1].Value.ToString();
    txtEmpPos.Text = this.dataGridView1.CurrentRow.Cells[2].Value.ToString();
    txtEmpAdd.Text = this.dataGridView1.CurrentRow.Cells[3].Value.ToString();

    txtUsername.Text = this.dataGridView1.CurrentRow.Cells[4].Value.ToString();
    txtPassword.Text = this.dataGridView1.CurrentRow.Cells[5].Value.ToString();
    txtEmail.Text = this.dataGridView1.CurrentRow.Cells[6].Value.ToString();
}

private void button8_Click(object sender, EventArgs e)
{
    adapt = new SqlDataAdapter("select * from Employee WHERE EmployeeName = '" + txtSearch.Text + "'", koneksi);
    dt = new DataTable();
    adapt.Fill(dt);
    dataGridView1.DataSource = dt;
}

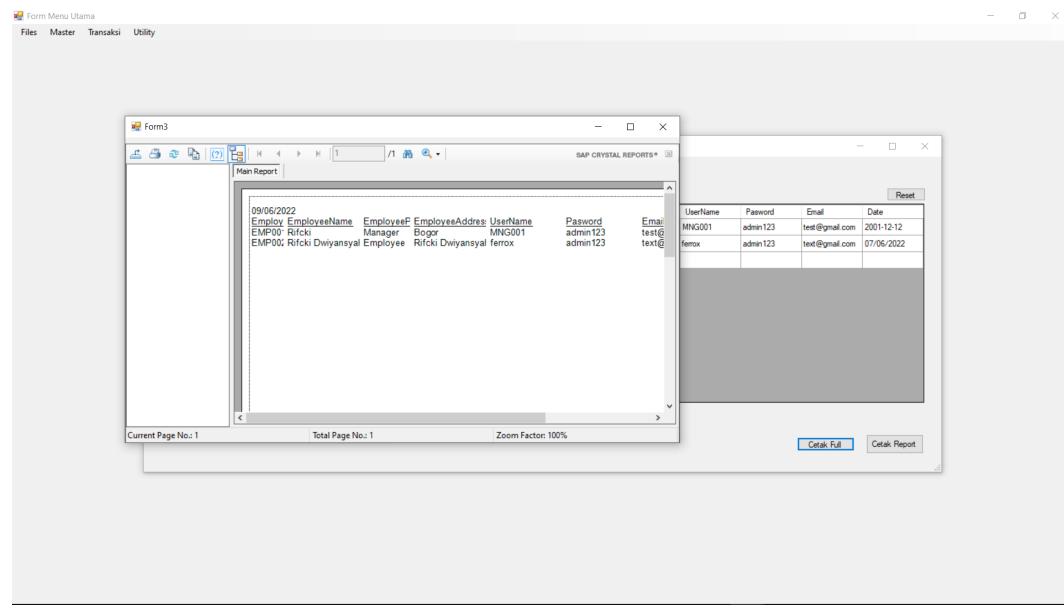
private void button9_Click(object sender, EventArgs e)
{
    showData();
}

private void button7_Click(object sender, EventArgs e)
{
    ReportViewerEmployee2 re = new ReportViewerEmployee2();
    re.Show();
}
}

```

Untuk code yang kami gunakan, ini sama dengan form yang ada di Form Obat, Customer, dan Form Supplier.

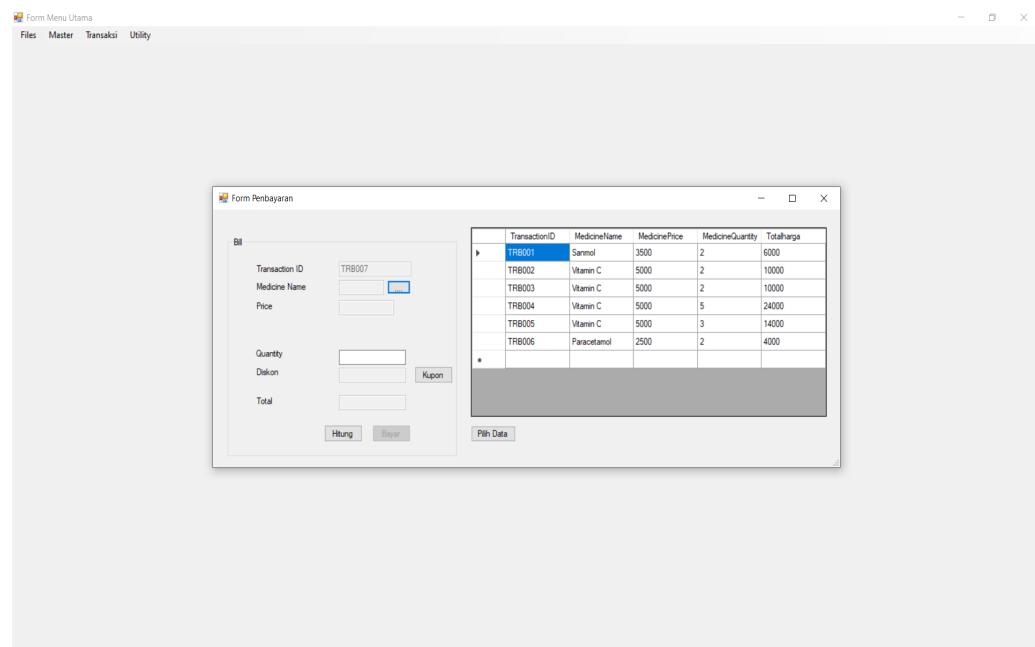
Jika menekan tombol report maka akan diarahkan ke List Employee dengan Crystal Report dibawah ini.



4.2.7. Transaksi

Dalam form Transaksi ini, dapat melihat data transaksi yang ada. Pada form transaksi ini terdapat form **Medicine** yang berguna untuk mengambil data medicine untuk transaksi. Berikut adalah beberapa button yang dapat membantu proses transaksi :

1. Kupon : untuk memberikan diskon dalam transaksi, Data kupon diambil dari notepad
2. Hitung : untuk menjumlahkan total transaksi yang dilakukan oleh customer
3. Bayar : untuk menyelesaikan transaksi dan data akan masuk ke database sebagai berhasil dibayar.



Berikut merupakan tahapan yang dilakukan dalam pembuatan **Form Transaksi**:

```

public partial class FormTransaksi : Form
{
    string koneksi = "Data Source=FERROX\SQLEXPRESS;Initial Catalog=UAS_Visprog;Integrated Security=True";
    DataSet ds = new DataSet();
    SqlCommand cmd;
    SqlDataAdapter adpt;
    DataTable dt;

    Koneksi konn = new Koneksi();
    private SqlDataAdapter da;
    private SqlDataReader rd;
    public FormTransaksi()
    {
        InitializeComponent();
    }

    int total = 0;
    int qty = 0;
    int price = 0;
    int diskon = 0;

    void showData()
    {
        adpt = new SqlDataAdapter("select * from Transaksi", koneksi);
        dt = new DataTable();
        adpt.Fill(dt);
        dataGridView1.DataSource = dt;
    }

    void NoOtomatis()
    {
        long hitung;
        string urutan;
        SqlDataReader rd;
        SqlConnection conn = konn.GetConn();
        conn.Open();
        cmd = new SqlCommand("SELECT TransactionID FROM Transaksi WHERE TransactionID IN(SELECT MAX(TransactionID) FROM Transaksi) ORDER BY TransactionID DESC", conn);
        rd = cmd.ExecuteReader();
    }
}

```

```

        rd.Read();
        if (rd.HasRows)
        {
            hitung = Convert.ToInt64(rd[0].ToString().Substring(rd["TransactionID"].ToString().Length - 3, 3)) + 1;
            string kodeUrutan = "000" + hitung;
            urutan = "TRB" + kodeUrutan.Substring(kodeUrutan.Length - 3, 3);
        }
        else
        {
            urutan = "TRB001";
        }
        rd.Close();
        txtTransID.Text = urutan;
        conn.Close();
    }

    private void label6_Click(object sender, EventArgs e)
    {

    }

    private void label7_Click(object sender, EventArgs e)
    {

    }

    private void label5_Click(object sender, EventArgs e)
    {

    }

    private void groupBox2_Enter(object sender, EventArgs e)
    {

    }

    private void FormTransaksi_Load(object sender, EventArgs e)
    {
        showData();
        NoOtomatis();
        dataGridView1.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill;
    }

    static string Diskon()
    {
        FileStream fs = new FileStream("D:\\Diskon.txt", FileMode.OpenOrCreate, FileAccess.Read, FileShare.ReadWrite);
        StreamReader sr = new StreamReader(fs);
        return (sr.ReadToEnd());
        sr.Close();
        fs.Close();
    }

    static void Penjualan(string data, int total )
    {
        FileStream fs = new FileStream("D:\\\\Penjualan.txt", FileMode.Append, FileAccess.Write, FileShare.ReadWrite);
        StreamWriter sv = new StreamWriter(fs);
        sv.WriteLine(data, total);
        sv.Close();
        fs.Close();
    }

    private void button3_Click(object sender, EventArgs e)
    {
        txtDisc.Text = Diskon();
    }

    private void button1_Click(object sender, EventArgs e)
    {
        Penjualan(txtTransID.Text, total);

        DataClasses1DataContext db = new DataClasses1DataContext();
        Transaksi cs = new Transaksi
        {
            TransactionID = txtTransID.Text,
            MedicineName = txtMedicineName.Text,
        };
    }
}

```

```

        MedicinePrice = (Convert.ToInt32(txtPrice.Text)),
        MedicineQuantity = (Convert.ToInt32(txtQty.Text)),
        Totalharga = total,
    };

    db.Transaksis.InsertOnSubmit(cs);
    try
    {
        db.SubmitChanges();
        MessageBox.Show("Input data berhasil");

        showData();
        txtTransID.Clear();
        txtMedicineName.Clear();
        txtPrice.Clear();
        txtQty.Clear();
        txtTotal.Clear();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.ToString());
    }
}

private void button4_Click(object sender, EventArgs e)
{
    if (txtDisc.Text == "")
    {
        diskon = 0;
    }
    else {
        diskon = Convert.ToInt32(txtDisc.Text);
    }

    price = Convert.ToInt32(txtPrice.Text);

    if(txtQty.Text == "")
        System.Windows.Forms.MessageBox.Show("Harap Isi Jumlah");
    qty = Convert.ToInt32(txtQty.Text);

    total = (price * qty) - diskon;
    txtTotal.Text = total.ToString();
    button1.Enabled = true;
}

private void button2_Click(object sender, EventArgs e)
{
    DaftarObat daftar = new DaftarObat();
    daftar.Show();
}

private void button5_Click(object sender, EventArgs e)
{
    DataClasses1DataContext db = new DataClasses1DataContext();
    TransactionDetail cs = new TransactionDetail
    {
        TransactionID = txtTransID.Text,
        MedicineID = txtMedID.Text,
        Quantity = txtQty.Text,
    };

    db.TransactionDetails.InsertOnSubmit(cs);
    try
    {
        db.SubmitChanges();
        MessageBox.Show("Input data berhasil");
    }
}

```

```

        showData();
        txtMedicineName.Text = "";

    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.ToString());
    }
}

private void button6_Click(object sender, EventArgs e)
{
    txtTransID.Text = this.dataGridView1.CurrentRow.Cells[0].Value.ToString();
    txtMedicineName.Text = this.dataGridView1.CurrentRow.Cells[1].Value.ToString();
    txtPrice.Text = this.dataGridView1.CurrentRow.Cells[2].Value.ToString();
    txtQty.Text = this.dataGridView1.CurrentRow.Cells[3].Value.ToString();
    txtTotal.Text = this.dataGridView1.CurrentRow.Cells[4].Value.ToString();
}

private void dataGridView1_CellContentClick(object sender, DataGridViewCellEventArgs e)
{
}
}

```

Untuk code yang kami gunakan pada form transaksi, kami mendisable semua textfield kecuali quantity, karena textfield akan terisi saat menekan button di sebelah Medicine Name. dimana saat kita menekan button tersebut nanti akan tampil pilihan obat yang akan dibeli. Saat kita memilih obatnya nanti textfield yang ada di dalam Form transaksi akan terisi.

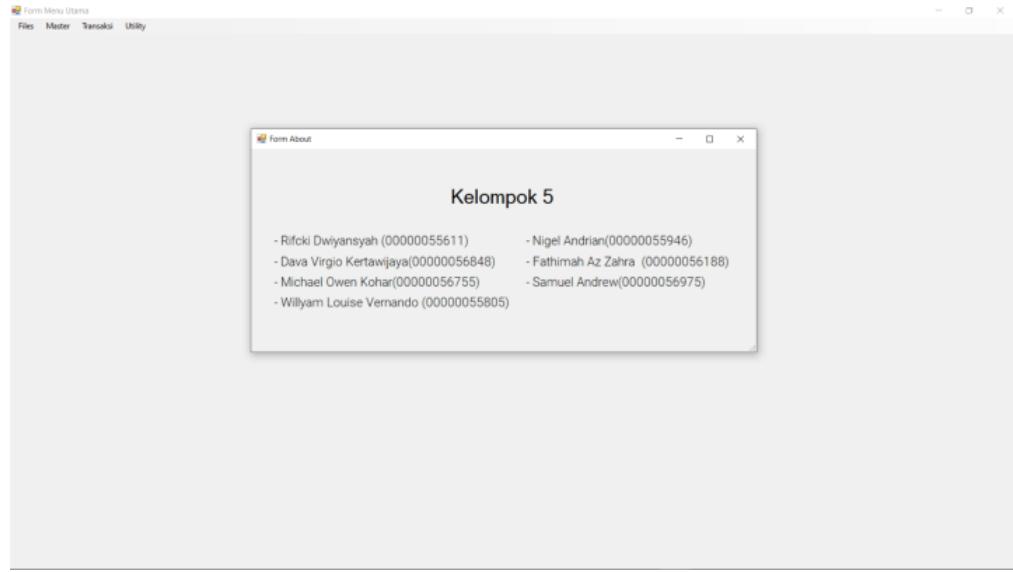
Lalu kami juga menggunakan fileStream untuk melakukan read and write file. dimana pada proses read, nanti program akan membaca sebuah file diskon/promo potongan harga. Lalu untuk writenya, program akan mendata semua transaksi yang terjadi dengan menampilkan ID Transaksi yang telah dibuat.

Dan juga pada form transaksi, sebelum kita melakukan pembayaran, kita harus menghitung total terlebih dahulu dengan menekan button hitung. dimana pada button hitung terdapat sebuah code yang menghitung total dengan hitungan $(\text{harga} * \text{qty}) - \text{diskon}$.

Lalu pada button bayar terdapat sebuah fungsi input data kedalam database dengan menggunakan LINQ.

4.2.8. Utility

Pada menu tampilan Utility terdapat sub menu ‘About’ yang berisi informasi mengenai pembuat aplikasi seperti yang terlihat pada tampilan di bawah ini.



Berikut codingan yang dibuat untuk membuat Form About

```
namespace Aplikasi_Project_Apotek_Kimia_Farma
{
    public partial class FormAbout : Form
    {
        public FormAbout()
        {
            InitializeComponent();
        }

        private void label2_Click(object sender, EventArgs e)
        {

        }

        private void label7_Click(object sender, EventArgs e)
        {
        }
    }
}
```

4.3. Query Pembuatan Tabel Basis Data

4.3.1. Create dan Use Database

```
Create database Final_VisProg
use Final_Visprog
```

4.3.2. Create Table Employee

```
CREATE TABLE Employee
(
    EmployeeID VARCHAR (8) PRIMARY KEY,
    EmployeeName VARCHAR (25),
    EmployeePosition VARCHAR (10)
        CHECK(EmployeePosition IN ('None', 'Karyawan', 'Manager'))
        DEFAULT 'None',
    EmployeeAddress VARCHAR (50),
    UserName VARCHAR (20),
    Password VARCHAR (40),
    Email VARCHAR (50),
    [Date] DATE,
);
```

4.3.3. Create Table Customer

```
CREATE TABLE Customer
(
    CustID VARCHAR (8) PRIMARY KEY,
    CustName VARCHAR (20),
    CustAddress VARCHAR (30),
    CustPhone VARCHAR (30),
);
```

4.3.4. Create Table Supplier

```
CREATE TABLE Suplier
(
    SuplierID VARCHAR (8) PRIMARY KEY,
    SuplierName VARCHAR (20),
    SuplierAddress VARCHAR (40),
    SuplierPhone VARCHAR (20),
);
```

4.3.5. Create Table TransactionHeader

```
CREATE TABLE TransactionHeader
(
    TransactionID CHAR (5) PRIMARY KEY,
    EmployeeID VARCHAR (8)
        FOREIGN KEY REFERENCES Employee (EmployeeID),
    SuplierID VARCHAR (8)
        FOREIGN KEY REFERENCES Suplier(SuplierID),
    CustID VARCHAR (8)
        FOREIGN KEY REFERENCES Customer (CustID),
    TransactionDate DATE NOT NULL,
);
```

4.3.6. Create Table Medicine

```
CREATE TABLE Medicine
(
    MedicineID VARCHAR (8) PRIMARY KEY,
    MedicineName VARCHAR (40) NOT NULL,
    MedicinePrice INT NOT NULL,
    MedicineDate DATE,
    MedicineQuantity INT NOT NULL
);
```

4.3.7. Create Table TransactionDetail

```
CREATE TABLE TransactionDetail
(
    TransactionID CHAR (5)
        FOREIGN KEY REFERENCES TransactionHeader(TransactionID),
    MedicineID VARCHAR (8)
        FOREIGN KEY REFERENCES Medicine(MedicineID),
    Quantity INT NOT NULL,
    PRIMARY KEY(TransactionID,MedicineID)
);
```

BAB V

KESIMPULAN

5.1. Kesimpulan

Dapat disimpulkan pembuatan kasir dengan memakai data apotek kimia farma, mampu diimplementasikan pembuatan database SQL Server dan Visual Programming dengan menggunakan .NET dengan menggunakan bahasa C# dapat mempermudah pengaksesan data apotek. Mencari Tanggal, Jadwal, Role dan lainnya didapat secara mudah dengan memasukkan data melalui aplikasi .NET yang telah dibuat. SQL sebagai wadah untuk membuat tabel database mampu menyambungkan koneksi kedalam aplikasi yang dibuat dengan bantuan crystalView dan juga menampilkannya dengan CrystalReport.

Peran Anggota

1. Dava Virgio Kertawijaya - (00000056848) : Membuat laporan, membuat latar belakang, tujuan, rumusan masalah, proses bisnis, masalah bisnis, dan penjelasan form dan kodingan dalam laporan, serta membantu ide perancangan aplikasi, membantu membuat form form aplikasi, dan lainnya.
2. Nigel Andrian - (00000055946) : Membuat Update data, Membantu membuat laporan, Penjelasan form pada laporan, Membantu membuat form employee, Rumusan Masalah, Proses Bisnis.
3. Michael Owen Kohar - (00000056755) : Membantu membuat laporan, Membantu membuat form employee, Membantu ide perancangan aplikasi, Membuat ERD, Membantu membuat Latar Belakang, Tujuan, Rumusan Masalah, Proses Bisnis, Masalah bisnis, Dan penjelasan Setiap Form dalam Laporan.
4. Samuel Andrew - (00000056975) : Membantu membuat Laporan, Merekam dan mengedit video presentasi aplikasi, Membantu ide perancangan aplikasi dan lainnya.
5. Rifcki Dwiyansyah - (00000055611) : Membuat CRUD Stok Obat, Membuat CRUD Employee, Membuat CRUD Customer, Membuat CRUD Supplier, Membantu CRUD, Transaksi, Membuat crystal report. Membuat fitur pilih data, Membuat Crystal Report, Menghubungkan Database ke Aplikasi. Pengujian Prototype Aplikasi. Membuat Login, Membuat Pilihan obat.
6. Fathimah Az Zahra - (00000056188) : Membantu pembuatan ERD, Tabel relasi, Normalisasi Database, Membuat bagian Kasir pada aplikasi, membantu pembuatan menu barang, membantu pembuatan insert pada database, membuat tabel menentukan isi pada Database, membuat aplikasi kasir, membantu membuat penjelasan aplikasi tentang fungsi - fungsi dari form, membuat crystal report dan membuat Enhanced Entity Relationship Diagram.
7. Willyam Louise Vernando - (00000055805) : Membuat Laporan bagian System Requirement, Membuat Normalisasi Database, Tabel Relasi, Enhanced Entity Relationship Diagram, Tabel Relasi, Penjelasan Aplikasi, Membuat Database Aplikasi, Membuat Aplikasi bagian Form Menu Utama, Menghubungkan Setiap Form ke Form Utama, AutoNumber, Search Bar, Merapikan DataGridView, Advance Crystal Report.

DATA DIRI MAHASISWA

NIM	Nama Lengkap	Program	No. HP	Email Student	Alamat Domisili
000000 56848	Dava Virgio Kertawijaya	Sistem Informasi	089527193296	dava.virgio@student.umn.ac.id	Palembang
000000 55946	Nigel Andrian	Sistem Informasi	082179588619	nigel.andrian@student.umn.ac.id	Palembang
000000 56755	Michael Owen Kohar	Sistem Informasi	08981040213	michael.owen@student.umn.ac.id	Palembang
000000 56975	Samuel Andrew	Sistem Informasi	088809623099	samuel.andrew@student.umn.ac.id	Jakarta
000000 55611	Rifcki Dwiyansyah	Sistem Informasi	081288762460	rifcki.dwiyansyah@student.umn.ac.id	Bogor
000000 56188	Fathimah Az Zahra	Sistem Informasi	081218818898	fathimah.az@student.umn.ac.id	Tangerang

000000	Willyam	Sistem	081273328008	willyam.louise	Palembang
55805	Louise	Informasi		@student.um n.ac.id	