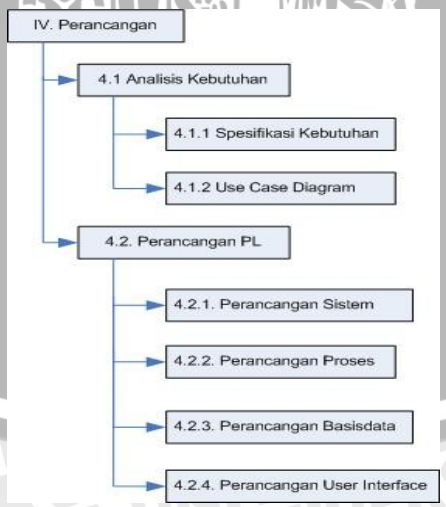
BAB IV
PERANCANGAN PERANGKAT LUNAK

Bab ini menjelaskan perancangan Sistem Transaksi Pembayaran *Food Court* dengan media *E-card* yang berbasis *barcode* dengan menggunakan kompiler Microsoft Visual Basic 6.0 dengan bahasa pemrograman basic dan database MySQL. Perancangan yang dilakukan meliputi dua tahap. Pada tahap pertama dilakukan proses analisis kebutuhan perangkat lunak dengan mengidentifikasi kebutuhan-kebutuhan yang diperlukan dalam pengembangan Sistem Transaksi Pembayaran *Food Court.* Kebutuhan-kebutuhan tersebut kemudian dimodelkan ke dalam suatu *use case diagram.* Pemodelan kebutuhan ditujukan untuk menggambarkan kebutuhan-kebutuhan yang harus disediakan oleh Sistem Transaksi Pembayaran *Food Court* agar dapat memecahkan permasalahan yang terjadi pada sebuah *Food Court.*

Sedangkan tahap yang kedua adalah proses perancangan (desain). Pada proses perancangan sendiri terdapat empat tahap yaitu : perancangan sistem, perancangan proses, perancangan basisdata, dan perancangan antarmuka / *User Interface* perangkat lunak. Gambar 4.1 menunjukkan langkah-langkah proses perancangan yang akan diulas pada bab ini.



Gambar 4.1 Diagram proses perancangan

Sumber : Perancangan

4.1 Analisis Kebutuhan Perangkat Lunak

Analisis kebutuhan perangkat lunak adalah aktifitas rekayasa perangkat lunak yang menjembatani antara kebutuhan ditingkat sistem dan perancangan perangkat lunak. Analisis kebutuhan adalah proses yang digunakan untuk mendapatkan, menganalisis, dan memvalidasi kebutuhan-kebutuhan sistem.

4.1.1 Spesifikasi Kebutuhan

Spesifikasi kebutuhan diperlukan untuk menjelaskan kebutuhan perangkat lunak yang telah didefinisikan sebelumnya secara lebih detail dan tepat yang akan menjadi dasar bagi perancangan dan implementasi.

Tabel 4.1 di bawah ini merupakan tabel spesifikasi kebutuhan dari Sistem Transaksi Pembayaran Food Court yang akan dibangun pada tugas akhir ini.

Tabel 4.1 Spesifikasi kebutuhan dari Sistem Transaksi Pembayaran Food Court

No	Kebutuhan	*Use Case*
1	Untuk memasukkan data pribadi *customer*, *password* serta pengaktivasian *E-card*	Registrasi
2	Untuk mengetahui sisa kredit yang terdapat pada *E-card*	Cek kredit
3	Untuk mengisi ulang kredit pada *E-card*	Refill kredit
4	Untuk melakukan pemesanan makanan pada *counter* makanan	Order makanan
5	Untuk melakukan pembayaran pada *counter* makanan	Pembayaran
6	Untuk mengetahui transaksi yang telah dilakukan oleh *customer*	Laporan transaksi
7	Untuk meng-*update* menu makanan	Update menu
8	Untuk mengubah status order *customer*	Ubah status
9	Untuk mengetahui rekapitulasi hasil penjualan	Laporan keuangan
10	Untuk mengetahui setiap transaksi yang telah terjadi pada *Food Court*	Monitoring transaksi

Sumber : Perancangan

4.1.2 Use Case Diagram

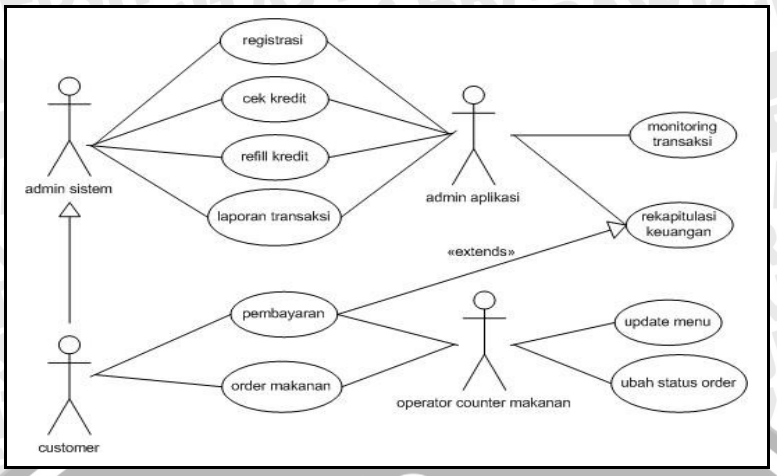
Dari rumusan kebutuhan – kebutuhan di atas, dimodelkan dalam *use case diagram*. Pada Tabel 4.2 dibawah ini memberikan uraian pada masing-masing aktor yang digunakan dalam diagram use case.

Tabel 4.2 Deskripsi actor yang terlibat dalam use case

Actor	Deskripsi *Actor*
Customer	Pengguna yang akan menggunakan sistem ini untuk melakukan order makanan dan pembayaran langsung pada counter makanan.
Admin Sistem	Pengguna yang akan menggunakan system ini untuk membantu customer melakukan registrasi, pengecekan kredit, dan refill kredit, serta laporan transaksi yang telah dilakukan oleh customer
Operator *Counter* makanan	Pengguna yang akan menggunakan sistem ini untuk melakukan update menu, melayani order, dan menerima pembayaran.
Admin Aplikasi	Pengguna yang akan menggunakan system ini untuk melakukan monitoring transaksi, dan rekapitulasi transaksi

Sumber: Perancangan

Pada gambar 4.2 dibawah ini diperlihatkan pemodelan dalam *diagram use case* yang menggambarkan interaksi yang dilakukan *customer*, admin sistem, admin aplikasi dan operator *counter* terhadap sistem.



Gambar 4.2 Use Case Diagram Sistem Transaksi Pembayaran Food Court
Sumber : Perancangan

4.2 Perancangan Perangkat Lunak

Perangkat Lunak Sistem Transaksi Pembayaran *Food Court* dibangun menggunakan kompiler *Microsoft Visual Basic 6.0* dengan bahasa pemrograman basic. Arsitektur dari perangkat lunak Sistem Transaksi Pembayaran *Food Court* adalah *Client-Server*, dimana sistem penyimpanan data dilakukan secara terpusat pada komputer *server*. *Database* yang nantinya digunakan untuk menyimpan data dari setiap transaksi yang terjadi pada *food court* adalah *Database* MySQL. *Database* MySQL dipilih sebagai media penyimpanan data karena kecepatan *query* yang tinggi dibandingkan dengan *database* lainnya. Perangkat lunak Sistem Transaksi Pembayaran *Food Court* menggunakan MyODBC sebagai *driver* penghubung untuk melakukan koneksi dengan *database* MySQL melalui ODBC yang terdapat pada sistem operasi.

Sistem Transaksi Pembayaran *Food Court* dirancang agar mampu :

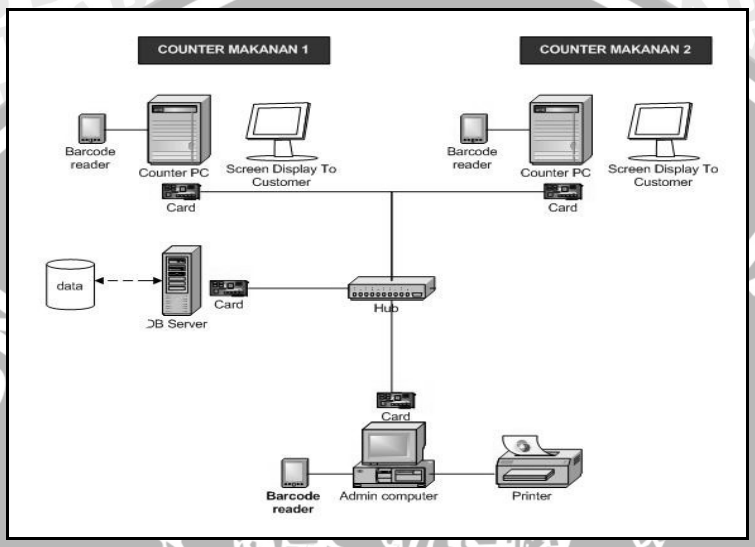
1. Melakukan pembayaran tanpa menggunakan uang tunai melainkan dengan menggunakan media *E-card* berbasis *barcode*.
2. Melakukan rekapitulasi laporan keuangan.

4.2.1 Perancangan Sistem

Perancangan sistem merupakan tahap awal dari perancangan perangkat lunak. Perancangan ini dilakukan untuk mengetahui kondisi sistem secara umum. Perancangan sistem meliputi diagram blok sistem dan diagram konteks.

4.2.1.1 Diagram Blok Sistem

Diagram blok sistem menggambarkan setiap blok atau bagian dari sistem aplikasi. Sistem Transaksi Pembayaran *Food Court* dirancang untuk dapat dijalankan pada jaringan lokal. Sistem aplikasi akan dibagi menjadi beberapa bagian yang terhubung dengan *server*, seperti yang diperlihatkan dalam Gambar 4.3 :



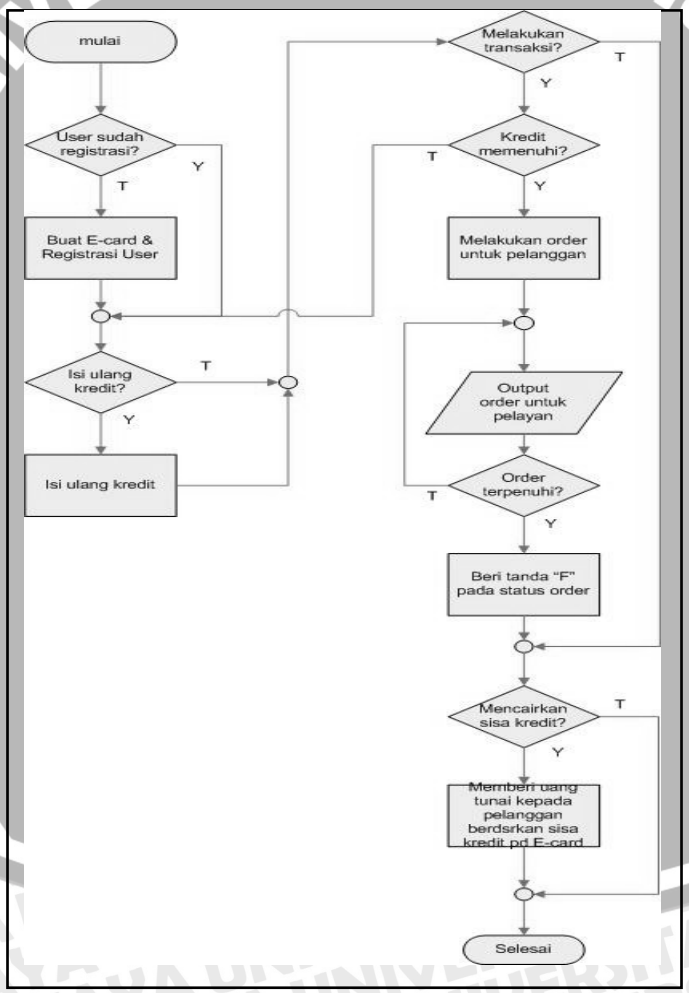
Gambar 4.3 Diagram Blok Sistem

Sumber : Perancangan

Prinsip Kerja Sistem

Sistem Transaksi Pembayaran *Food Court* terdiri dari *hardware* dan *software* yang saling mendukung. Hardware yang digunakan yaitu *barcode reader*, berfungsi sebagai alat untuk menkodekan *barcode* yang terdapat pada *E-card*, dimana pada *barcode* terdapat nomor unit yang akan mereferensikan database pelanggan. Prinsip kerja dari sistem ini diawali dari proses registrasi *customer* yang belum memiliki *E-card*. Pada saat proses registrasi, *E-card* akan diaktivasi dengan penggunaan *password* oleh *customer*, yang juga berfungsi sebagai pengaman untuk menghindari pemakaian oleh orang yang tidak berhak. Setelah pengaktivasian, *customer* dapat menggunakan *E-card* pada *counter-counter* makanan yang tersedia pada *food court* yang menggunakan sistem tersebut.

Bagi *customer* yang telah memiliki *E-card* atau sudah melakukan registrasi, maka dapat dilakukan pengecekan sisa kredit dan melakukan pengisian ulang kredit tersebut. Pada saat *customer* melakukan order makanan, transaksi pembayaran langsung dilakukan pada *counter* makanan tersebut. Operator *counter* akan membaca *barcode* pada *E-card* melalui *barcode reader* dan melihat sisa kredit serta transaksi yang telah dilakukan. Pada saat *customer* menunggu pesanan, Operator *counter* dapat melihat status order pelanggan yang selanjutnya akan menginformasikan kepada pelayan Jika diinginkan, *customer* dapat mencairkan sisa kredit yang ada pada *E-card* tersebut setelah *customer* melakukan transaksi pembayaran. Proses yang terjadi dalam sistem diperlihatkan pada gambar 4.4 :

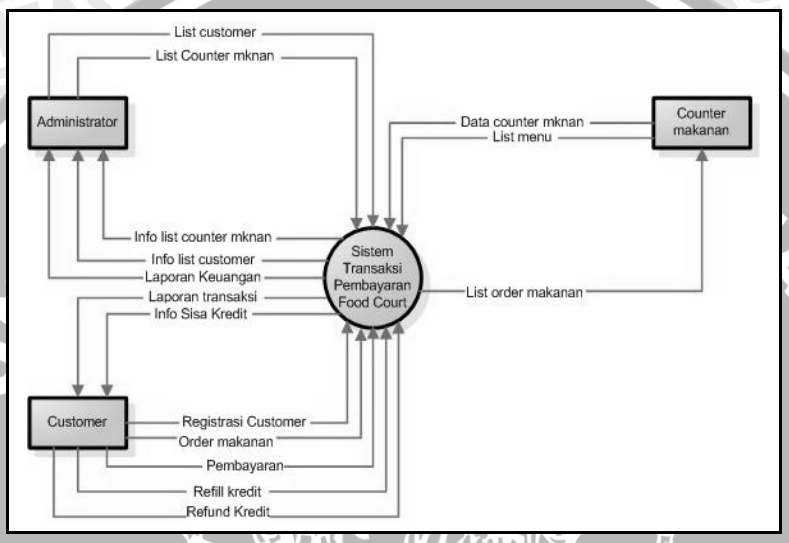


Gambar 4.4 Flowchart Kerja Sistem
Sumber : Perancangan

Diagram Konteks

DFD yang pertama kali dibuat adalah diagram konteks (*context diagram*). Diagram konteks menggambarkan seluruh input ke sistem atau output dari sistem serta menjelaskan hubungan sistem dengan lingkungan atau kesatuan luar. Pada sistem ini, melibatkan 3 kesatuan luar, yaitu administrator, *customer,*dan *counter* makanan.

Diagram konteks ditunjukkan pada gambar berikut :



Gambar 4.5. Diagram Konteks Sistem Transaksi Pembayaran Food Court

Sumber : Perancangan

Berdasarkan Gambar 4.5 pada proses sistem transaksi pembayaran pada sebuah *food court* mempunyai data masukan berupa :

1. *Registrasi Customer* [IRS_2110]

   Registrasi *customer* merupakan proses pendaftaran *customer* dengan memasukkan data pribadi berdasarkan KTP/SIM, dan *password* untuk pengaktivasian *E-card*.

   Parameter yang digunakan :

   - *customer_ID* [IRS_2111]: berisi masukan tertentu berdasarkan barcode yang terdapat pada *E-card*.
   - *Password* [IRS_2112]: berisi masukan *password* yang telah ditentukan oleh *customer*.

2. *Order Makanan* [IRS_2120]

   Order Makanan digunakan untuk proses pemesanan makanan.

   Parameter yang digunakan :

- *food_ID* [IRS_2121]: berisi masukan berupa ID yang mewakili nama makanan yang dipesan.
- *counter_ID* [IRS_2122]: berisi masukan berupa ID *counter*, tempat dimana order makanan dilakukan.

3. *Pembayaran* [IRS_2130]

   Pembayaran digunakan untuk proses pembayaran dari makanan yang telah dipesan oleh customer.

   Parameter yang digunakan :

   - *trans_ID* [IRS_2131] : berisi masukan tertentu yang ditentukan oleh sistem
   - *customer_ID* [IRS_2132] : berisi masukan tertentu berdasarkan *barcode* yang terdapat pada *E-card*.

4. *Refill Credit* [IRS_2140]

   Refill Credit digunakan untuk proses pengisian ulang kredit yang terdapat pada *E-card*.

   Parameter yang digunakan :

   - *customer_ID* [IRS_2141] : berisi masukan tertentu berdasarkan *barcode* yang terdapat pada *E-card*.

5. *Refund Credit* [IRS_2150]

   Refund Credit digunakan untuk proses pencairan kredit yang terdapat pada E-card dalam bentuk uang tunai.

   Parameter yang digunakan :

   - Customer_ID [IRS_2151] : berisi masukan tertentu berdasarkan *barcode* yang terdapat pada *E-card*.

6. *Data counter makanan* [IRS_2160]

   Data *counter* makanan berupa nama *counter* dan data pemilik *counter*.

   Parameter yang digunakan :

   - *counter_ID* [IRS_2161] : berisi masukan berupa ID *counter* yang telah ditentukan oleh sistem.

7. *List Menu* [IRS_2170]

   List Menu berupa daftar menu yang dimiliki oleh setiap *counter* makanan.

   Parameter yang digunakan :

- *food_ID* [IRS_2171] : berisi masukan berupa ID yang mewakili nama makanan yang disediakan oleh *counter* makanan.

Berdasarkan Gambar 4.5 pada proses sistem transaksi pembayaran pada sebuah *food court* mempunyai data keluaran berupa :

1. *Info list customer* [IRS_2180]

   Berisi informasi mengenai *customer* yang telah terdaftar dan disimpan dalam *database*.

2. *Info list counter makanan* [IRS_2190]

   Berisi informasi mengenai *counter* makanan yang telah terdaftar dan disimpan dalam *database*.

3. *Info list menu* [IRS_2200]

   Berisi informasi mengenai menu dari setiap *counter* makanan.

4. *Info sisa kredit* [IRS_2210]

   Berisi informasi mengenai sisa kredit yang terdapat pada *E-card customer*.

5. *List order makanan* [IRS_2220]

   Berisi informasi mengenai makanan yang telah dipesan oleh *customer*.

6. *Laporan transaksi* [IRS_2230]

   Berisi informasi mengenai transaksi yang telah dilakukan oleh *customer*.

7. *Laporan keuangan* [IRS_2240]

   Berisi informasi mengenai pembagian hasil antara pihak *counter* makanan dan pihak pengelola berdasarkan transaksi yang telah terjadi pada masing-masing *counter*.

4.2.1.3 Rancangan *Barcode* pada *E-card*

Sistem Transaksi Pembayaran *Food Court* menggunakan salah satu teknologi *E-card* yaitu *barcode* dengan jenis *Code 128*. *Barcode Code 128* dipilih karena memiliki kerapatan tinggi, dapat mengkodekan keseluruhan simbol ASCII (128 karakter) dalam luasan yang paling minim dibandingkan dengan *barcode* jenis lain. *Code 128* menggunakan 4 ketebalan elemen (bar dan spasi) yang

berbeda dengan *barcod*e jenis lain yang kebanyakan menggunakan 2 ketebalan elemen.

Setiap karakter pada *code 128* dikodekan oleh 3 bar dan 3 spasi (atau 6 elemen) dengan ketebalan masing-masing elemen 1 sampai 4 kali ketebalan minimum (*module*), jika dihitung dngan satuan *module* maka tiap karakter *code 128* terdiri dari 11 *module* kecuali untuk *stop character* yang terdiri dari 4 bar 3 spasi (13 *module*). Jumlah total *module* untuk bar selalu genap sedangkan untuk spasi selalu ganjil, selain itu *code 128* memiliki 3 *start character* yang berbeda sehingga code 128 memiliki 3 sub set karakter yang bersesuaian *dengan start character*-nya seperti tampak pada Tabel 4.3.

Code 128 memiliki fitur untuk dapat bergeser dari *subset* yang satu ke *sub set* yang lainnya dengan menggunakan karakter CODE dan SHIFT, CODE X menyebabkan seluruh *message* bergeser menjadi *sub set* X (misalnya CODE A pada *sub set* B membuat *message* menjadi sub set A), sedangkan SHIFT menyebabkan satu karakter didepannya bergeser *sub set* (ini hanya berlaku untuk *sub set* A ke *sub set* B atau sebaliknya). Struktur *code 128 barcode* seperti terlihat dibawah ini :

Gambar 4.6. Struktur Code 128

Sumber : *Anonymous* , 2006 : 4

dimana tinggi *barcode* minimum 0.15 kali lebar *barcode* dan lebar *barcode* dinyatakan dalam rumus :

L = (11C+35)X untuk alphanumeric (CODE A dan CODE B)
L = (5.5C+35)X untuk double density numeric only (CODE C)

Dimana :

L : Lebar *barcode* total termasuk *quiet zone*

C : Jumlah karakter

X : Lebar *module* (elemen yang tersempit)

CODE C dikatakan sebagai *double densisty numeric only* dan dalam perhitungan lebar per karakternya hanya 5.5X sebab satu karakter CODE C mewakili 2 digit *numeric* (lihat tabel diatas).

Perhitungan *check character code 128* sebagai berikut :

1. Message : CODE128
2. Karakter : StartA C O D E 1 2 8
3. Nilai karakter : 103 35 47 36 37 0 17 18 24
4. Posisi : - 1 2 3 4 5 6 7 8
5. Perhitungan Total : 103 + (35 x 1) + (47 x 2) + (36 x 3) + (37 x 4) + (0x5) + (17x6) + (18x7) + (24 x 8) = 908

   908 / 103 = 8 *remainder* 84

   84 =DC4
6. Message akhir : (StartA)CODE128(DC4)(STOP)

Konfigurasi *Barcode* pada *E-card* yang diterapkan dalam Sistem Transaksi Pembayaran *Food Court* diperlihatkan dalam Gambar 4.8

Gambar 4.7 Sampel Barcode pada E-card

Sumber : Perancangan

Pada Gambar 4.7 memiliki 2 informasi, yaitu 080901 yang memuat informasi mengenai tanggal pembuatan kartu dengan format YYMMDD dan 0001 yang memuat urutan kartu berdasarkan banyaknya kartu yang dibuat pada tanggal tersebut .

Table 4.3. Tabel karakter set Code 128

Nilai	Karakter Set			Encoding	Nilai	Karakter Set			Encoding
	A	B	C			A	B	C	
00	SP	SP	00	11011001100	53	U	U	53	11011101110
01	!	!	01	11001101100	54	V	V	54	11101011000
02	"	"	02	11001100110	55	W	W	55	11101000110
03	#	#	03	10010011000	56	X	X	56	11100010110
04	$	$	04	10010001100	57	Y	Y	57	11101101000
05	%	%	05	10001001100	58	Z	Z	58	11101100010
06	&	&	06	10011001000	59	[	[	59	11100011010
07	'	'	07	10011000100	60	\	\	60	11101111010
08	(	(	08	10001100100	61	]	]	61	11001000010
09	)	)	09	11001001000	62	^	^	62	11110001010
10	*	*	10	11001000100	63	_	_	63	10100110000
11	+	+	11	11000100100	64	NUL	`	64	10100001100
12	,	,	12	10110011100	65	SOH	a	65	10010110000
13	-	-	13	10011011100	66	STX	b	66	10010000110
14	.	.	14	10011001110	67	ETX	c	67	10000101100
15	/	/	15	10111001100	68	EOT	d	68	10000100110
16	0	0	16	10011101100	69	ENQ	e	69	10110010000
17	1	1	17	10011100110	70	ACK	f	70	10110000100
18	2	2	18	11001110010	71	BEL	g	71	10011010000
19	3	3	19	11001011100	72	BS	h	72	10011000010
20	4	4	20	11001001110	73	HT	i	73	10000110100
21	5	5	21	11011100100	74	LF	j	74	10000110010
22	6	6	22	11001110100	75	VT	k	75	11000010010
23	7	7	23	11101101110	76	FF	l	76	11001010000
24	8	8	24	11101001100	77	CR	m	77	11110111010
25	9	9	25	11100101100	78	SO	n	78	11000010100
26	:	:	26	11100100110	79	SI	o	79	10001111010
27	;	;	27	11101100100	80	DLE	p	80	10100111100
28	<	<	28	11100110100	81	DC1	q	81	10010111100
29	=	=	29	11100110010	82	DC2	r	82	10010011110
30	>	>	30	11011011000	83	DC3	s	83	10111100100
31	?	?	31	11011000110	84	DC4	t	84	10011110100
32	@	@	32	11000110110	85	NAK	u	85	10011110010
33	A	A	33	10100011000	86	SYN	v	86	11110100100
34	B	B	34	10001011000	87	ETB	w	87	11110010100
35	C	C	35	10001000110	88	CAN	x	88	11110010010
36	D	D	36	10110001000	89	EM	y	89	11011011110
37	E	E	37	10001101000	90	SUB	z	90	11011110110
38	F	F	38	10001100010	91	ESC	{	91	11110110110
39	G	G	39	11010001000	92	FS	|	92	10101111000
40	H	H	40	11000101000	93	GS	}	93	10100011110
41	I	I	41	11000100010	94	RS	~	94	10001011110
42	J	J	42	10110111000	95	US	DEL	95	10111101000
43	K	K	43	10110001110	96	FNC3	FNC3	96	10111100010
44	L	L	44	10001101110	97	FNC2	FNC2	97	11110101000
45	M	M	45	10111011000	98	SHIFT	SHIFT	98	11110100010
46	N	N	46	10111000110	99	Code C	Code C	99	10111011110
47	O	O	47	10001110110	100	Code B	FNC4	Code B	10111101110
48	P	P	48	11101110110	101	FNC4	Code A	Code A	11101011110
49	Q	Q	49	11010001110	102	FNC1	FNC1	FNC1	11110101110
50	R	R	50	11000101110	103	START A	START A	START A	11010000100
51	S	S	51	11011101000	104	START B	START B	START B	11010010000
52	T	T	52	11011100010	105	START C	START C	START C	11010011100
						STOP	STOP	STOP	11000111010

Sumber : *Anonymous* , 2006 : 4

4.2.2 Perancangan Proses

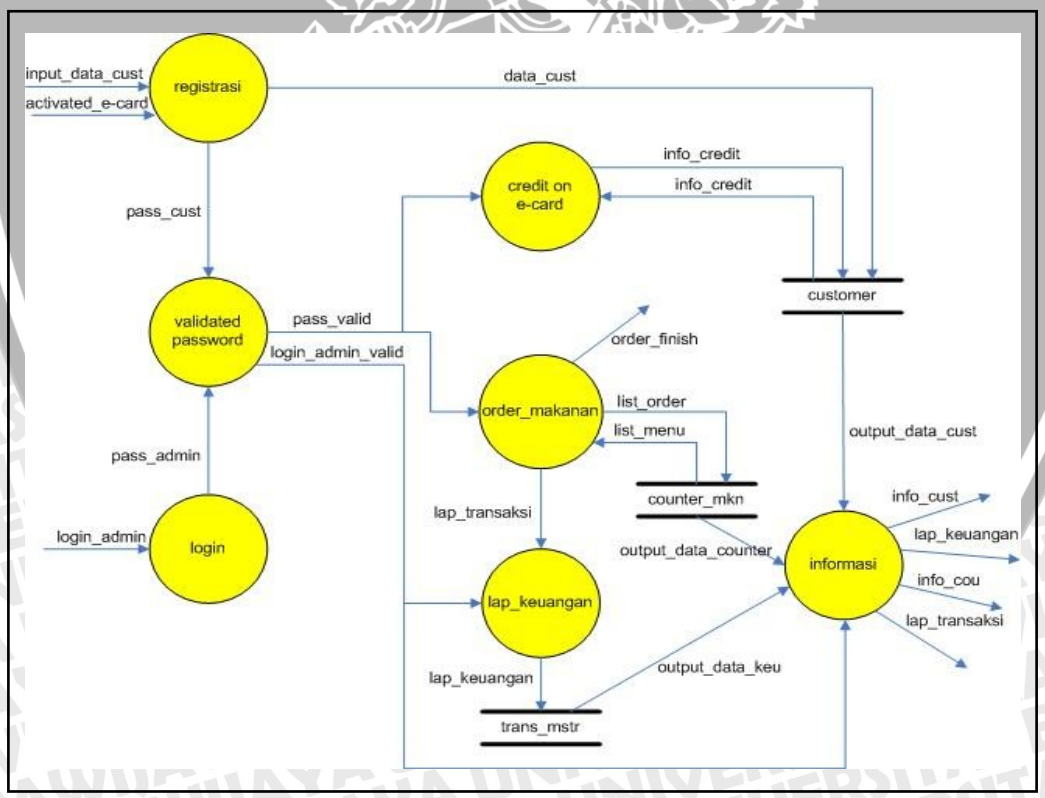
Perancangan proses menjelaskan masukan dan keluaran dari setiap proses yang terjadi pada Sistem Transaksi Pembayaran Food Court. Perancangan proses yang dilakukan dengan membuat *Data Flow Diagram* (DFD).

4.2.2.1 Data Flow Diagram

Data Flow Diagram (DFD) merupakan salah satu alat yang digunakan untuk menggambarkan keseluruhan Sistem Transaksi Pembayaran Food Court yang akan dibuat. DFD yang pertama kali dibuat adalah DFD level 0 atau *Context Diagram.* DFD *context diagram* telah dijelaskan sebelumnya pada perancangan sistem dengan bahasan diagram konteks.

4.2.2.1.1 DFD level 0

Tahap selanjutnya yang merupakan penjabaran diagram konteks adalah pembuatan DFD level 0.



Gambar 4.8 DFD Level 0
Sumber : Perancangan

Gambar 4.8 merupakan DFD level 0 dari Sistem Transaksi Pembayaran *Food Court* yang mempunyai tujuh proses, yaitu : registrasi, validasi password, login, kredit pada *E-card*, order makanan, laporan keuangan, dan informasi. Untuk detailnya maka dapat dijabarkan proses-proses tersebut sebagai berikut :

1. Proses *Registrasi* (proses 1)

   Terdiri dari dua masukan yaitu input data *customer* dan pengaktivasian *E-card*. Input data *customer* berupa data pribadi *customer* berdasarkan KTP/SIM sedangkan pengaktivasian *E-card* berupa *password* yang ditentukan oleh *customer*. Selanjutnya data pribadi *customer* akan disimpan pada tabel *customer*. Proses ini mempunyai data masukan :

   a. *input_data_cust* [IRS_2250]
   b. *activated_e-card* [IRS_2260]
      customer menentukan password untuk aktivasi E-card

   Proses ini mempunyai data keluaran

   a. *pass_cust* [IRS_2270]
   b. *data_cust* [IRS_2280]

2. Proses login (proses 2)

   Pada proses ini admin melakukan login untuk bisa masuk ke proses berikutnya. Nilai login yang dimasukkan berupa *employee_ID* dan *password*.
   Proses ini mempunyai data masukan:

   a. *login_admin* [IRS_2290]

   Proses ini mempunyai data keluaran:

   a. *pass_admin* [IRS_2300]

3. Proses *Validasi Password* (proses 3)

   Terdiri dari dua masukan berupa *pass_cust* dan *pass_admin*. Pada proses ini masukkan dari *password* akan diproses untuk mengetahui status aktivasi *E-card* dari *customer*. Proses validasi *password* ini mempunyai data masukkan:

   a. *pass_cust* [IRS_2270]

b. *pass_admin* [IRS_2300]

Proses ini mempunyai data keluaran berupa

a. *pass_valid* [IRS_2310]

jika *customer* berhasil memasukkan *password* yang benar.

b. *login_admin_valid* [IRS_2320]

jika *login admin* berhasil melakukan *validasi login*

4. Proses *credit on E-card* (proses 4)

Pada proses ini *customer* dapat melakukan pengecekan sisa kredit, pengisian ulang kredit, dan pencairan kredit pada *E-card*. *info_credit* merupakan hasil keluaran yang berupa teks yang disimpan pada tabel *customer*. Proses ini mempunyai data masukan berupa :

a. *pass_valid* [IRS_2310]

jika password yang dimasukkan benar.

Proses ini menggunakan *data store* :

a. *info_credit* [IRS_2330]

digunakan untuk menyimpan data mengenai kredit pada E-card yang dimiliki *customer*.

5. Proses *order_makanan* (proses 5)

Pada proses ini *customer* melakukan pemesanan makanan dan pembayaran.

Proses ini mempunyai data masukkan berupa :

a. *pass_valid* [IRS_2310]

jika *password* yang dimasukan benar.

Proses ini mempunyai data keluaran berupa :

α. *order_finish* [IRS_2340]

jika order dari *customer* telah terpenuhi.

β. *list_order* [IRS_2350]

berupa list makanan yang dipesan oleh *customer*.

Proses ini menggunakan *data store*:

a. *list_menu* [IRS_2360]

digunakan untuk menyimpan list data menu yang disediakan oleh counter makanan yang terdapat pada server.

6. Proses *lap_keuangan* (proses 6)

Pada proses ini dilakukan rekapitulasi terhadap transaksi pembayaran yang telah terjadi pada food court, sekaligus perhitungan bagi hasil antara pihak pengelola food court dengan pihak counter. Selain itu, pada proses ini customer dapat mengetahui banyaknya transaksi yang telah dilakukan. Data keluaran dari proses ini akan disimpan dalam tabel *trans_mstr*. Fasilitas ini hanya dapat digunakan oleh admin aplikasi.

Proses ini mempunyai data masukan berupa :

a. *login_admin_valid* [IRS_2320]
   jika admin berhasil melakukan *login*.
b. *lap_transaksi* [IRS_2370]
   berupa rekapan transaksi yang dilakukan *customer*.

Proses ini mempunyai data keluaran berupa:

a. *lap_keuangan* [IRS_2380]
   berupa rekapitulasi keuangan dari setiap transaksi yang telah terjadi pada tiap *counter* makanan.

7. Proses informasi (proses 7)

Terdiri dari tiga masukan yaitu berupa *output_data_cust*, *output_data_keu*, dan *output_data_counter*.

Proses ini mempunyai masukan data berupa :

a. *login_admin_valid* [IRS_2320]
   jika login admin berhasil maka proses informasi akan berjalan.

Proses ini mempunyai data keluaran :

a. *info_cust* [IRS_2390]
   berisi data-data customer.
b. *info_counter* [IRS_2400]
   berisi data-data counter makanan
c. *lap_keuangan* [IRS_2380]

berisi rekapitulasi keuangan dari setiap transaksi yang telah terjadi pada tiap *counter* makanan.

d. *lap_transaksi* [IRS_2370]

berisi laporan dari setiap transaksi yang telah dilakukan oleh customer.

Proses ini mempunyai *data store*:

a. *customer* [IRS_2410]

digunakan untuk menyimpan data-data dari customer

b. *counter_mkn* [IRS_2420]

digunakan untuk menyimpan data-data mengenai counter makanan

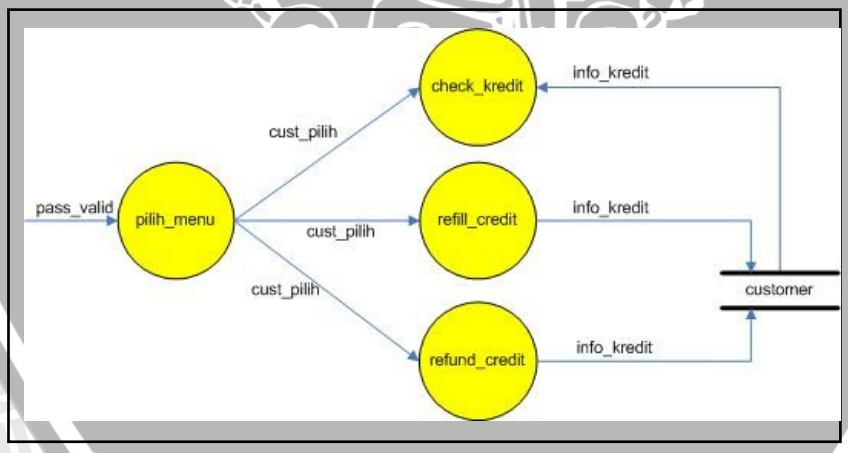
c. *trans_mstr* [IRS_2430]

digunakan untuk menyimpan data-data dari setiap transaksi yang telah terjadi.

4.2.2.1.2 DFD level 1

DFD Level 1 merupakan diagram yang menguraikan proses pada DFD level 0 yang mempunyai rincian proses. Pada sistem ini, mempunyai rincian proses yang terdiri dari satu sub proses.

4.2.2.1.2.1 DFD level 1 Proses Credit on E-card



Gambar 4.9 DFD Level 1 Proses Credit on E-card

Sumber : Perancangan

Gambar 4.9 menunjukkan perincian proses credit on E-card menjadi empat proses yang saling berhubungan, yaitu : pilih_menu, check_credit, refill_credit, dan refund_credit.

1. Proses *pilih_menu* (Proses 4.1)

   Merupakan proses untuk memilih aksi yang akan dilakukan terhadap kredit

pada E-card.

Proses ini mempunyai masukan berupa:

a. *pass_valid* [IRS_2310]

   jika *customer* berhasil memasukkan *password* yang benar

Proses ini mempunyai keluaran berupa :

a. *cust_pilih* [IRS_2440]

2. Proses *check_credit* (Proses 4.2)

   Merupakan proses untuk melakukan pengecekan sisa kredit yang terdapat pada E-card

   Proses ini mempunyai masukan berupa :

   a. *cust_pilih* [IRS_2440]

   Proses ini menggunakan *data store* :

   a. *info_credit* [IRS_2330]

      digunakan untuk menyimpan data mengenai banyaknya kredit yang dimiliki *customer*.

3. Proses *refill_credit* (Proses 4.3)

   Merupakan proses untuk melakukan pengisian ulang kredit yang terdapat pada E-card.

   Proses ini mempunyai masukan berupa :

   a. *cust_pilih* [IRS_2440]

   Proses ini mempunyai keluaran berupa :

   a. *info_credit* [IRS_2330]

      digunakan untuk menyimpan data mengenai banyaknya kredit yang dimiliki *customer*.

4. Proses *refund_credit* (Proses 4.4)

   Merupakan proses untuk melakukan pencairan kredit yang terdapat pada E-card dalam bentuk uang tunai.

   Proses ini mempunyai masukan berupa :

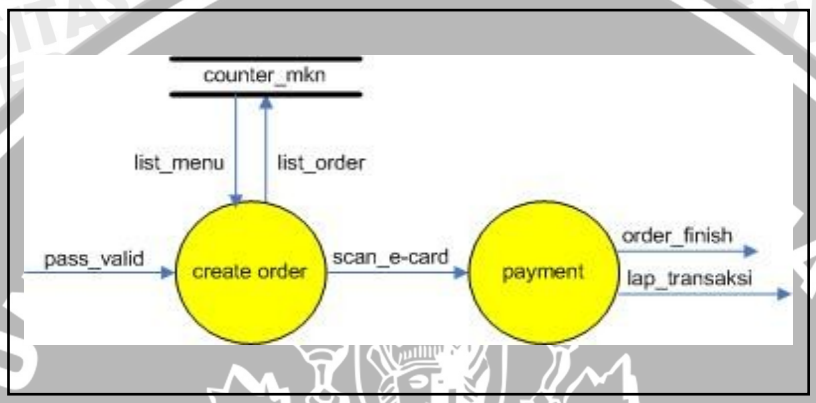
   a. *cust_pilih*[IRS_2440]

Proses ini mempunyai keluaran berupa :

a. *info_credit* [IRS_2330]

digunakan untuk menyimpan data mengenai banyaknya kredit yang dimiliki *customer*.

4.2.2.1.2.2 DFD level 1 Proses Order Makanan



Gambar 4.10 DFD Level 1 Proses Order Makanan

Sumber : Perancangan

Gambar 4.10 menunjukkan perincian proses order makanan menjadi dua proses yang saling berhubungan, yaitu create_order dan payment.

1. Proses *create_order* (proses 5.1)

   Merupakan proses untuk melakukan pemesanan makanan yang dilakukan pada masing-masing counter.

   Proses ini mempunyai masukan berupa :

   a. *pass_valid* [IRS_2310]

      jika *customer* berhasil memasukkan *password* yang benar.

   Proses ini mempunyai keluaran berupa :

   a. *list_order* [IRS_2350]

      berupa list makanan yang dipesan oleh *customer*.

   Proses ini menggunakan *data store*:

   c. *list_menu* [IRS_2360]

      digunakan untuk menyimpan list data menu yang disediakan oleh counter makanan yang terdapat pada server.

2. Proses *payment* (proses 5.2)

Merupakan proses untuk melakukan pembayaran dengan menggunakan E-card, dimana barcode scanner akan membaca kode-kode bar pada E-card untuk mengetahui customer_ID dan melakukan pengurangan kredit yang tersedia pada E-card berdasarkan billing dari transaksi yang telah dilakukan .

Proses ini mempunyai masukan berupa :

a. *scan_E-card* [IRS_2450]
   untuk mengetahui customer_ID dan melakukan pengurangan kredit yang tersedia pada E-card.

Proses ini mempunyai keluaran berupa :

a. *lap_transaksi* [IRS_2370]
   berupa rekapan transaksi yang telah dilakukan oleh *customer* dan laporan tersebut akan tersimpan pada tabel .
b. *order_finish* [IRS_2340]
   jika order dari *customer* telah terpenuhi.

4.2.3 Perancangan *database*

Perancangan *database* dilakukan agar Sistem Transaksi Pembayaran Food Court menjadi *database* yang efisien dalam penggunaan ruang penyimpanan, cepat dalam pengaksesan dan mudah dalam pemanipulasian data. Perancangan *database* dapat dilakukan dengan menggunakan *Entity Relationship Diagram* (Diagram ER), normalisasi data, dan *Data Object Descripiton*

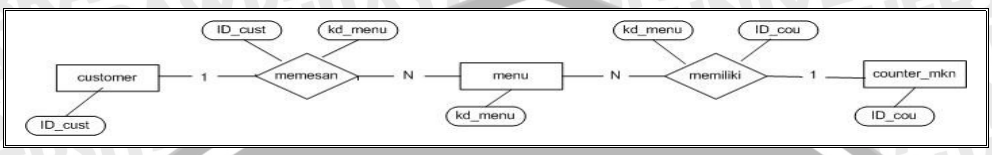
4.2.3.1 Entity Relationship Diagram (Diagram ER)

Diagram ER digunakan untuk menggambarkan entitas-entitas secara umum yang terdapat pada Sistem Transaksi Pembayaran *Food Court*. Akan tetapi perancangan database dengan diagram ER tidak sepenuhnya mewakili struktur yang terdapat pada keadaan nyata dari Sistem Transaksi Pembayaran *Food Court* .

Diagram ER akan menggambarkan hubungan entitas satu dengan lainnya dengan memperlihatkan hubungan antar atribut yang akan dijadikan *key* untuk berelasi antar tabel. Sebelum menggambarkan relasi diagram ER, entitas-entitas pembentuk sistem harus ditentukan terlebih dahulu beserta atribut entitas itu

sendiri. Entitas yang ditentukan merupakan tabel yang akan dipakai pada database.

Diagram ER dari struktur *database* `foodcourt` digambarkan dalam Gambar 4.11 berikut:

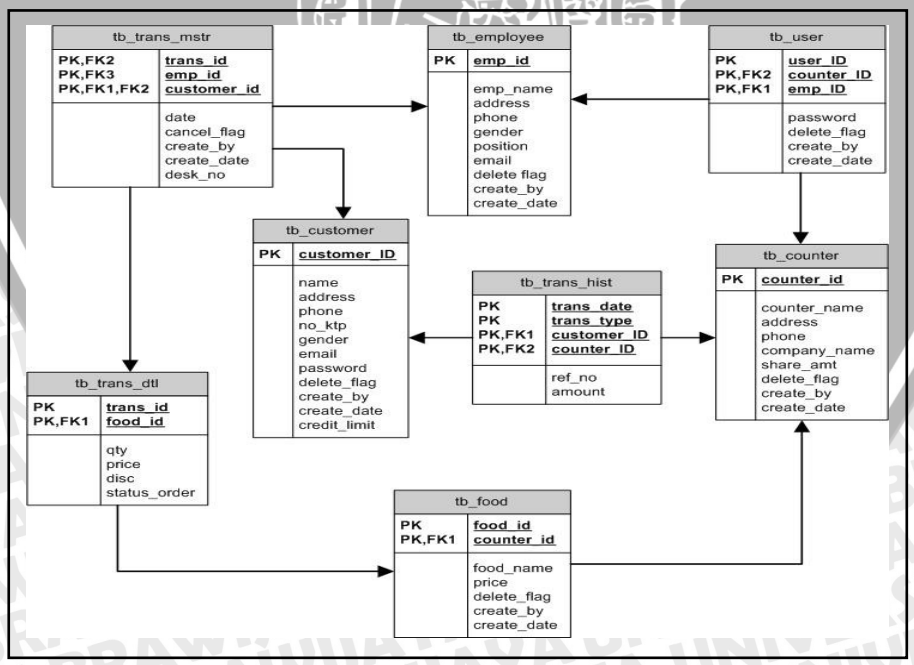


Gambar 4.11 Diagram ER pada *database* `foodcourt`

Sumber : Perancangan

4.2.3.2 Normalisasi Data

Pada Sistem Transaksi Pembayaran Food Court, normalisasi data dibutuhkan untuk mengurangi pengulangan (redundansi) data di setiap tabel dan mempercepat proses pencarian data di dalam *database*. Tabel-tabel yang akan digunakan untuk menyimpan data dinormalisasi menjadi enam buah entitas (tabel) yaitu Tabel `tb_customer`, `tb_counter`, `tb_food`, `tb_employee`, `tb_trans_mstr`, `tb_trans_dtl`, `tb_trans_hist`, dan `tb_user`. Rancangan normalisasi tabel-tabel tersebut diperlihatkan dalam Gambar 4.12 berikut:



Gambar 4.12 Normalisasi pada database `foodcourt`

Sumber : Perancangan

Tabel-tabel dalam *database* food_court dirancang seperti dalam Gambar 4.13 dengan tujuan mempercepat proses pencarian data di dalam masing-masing tabel. Untuk proses pencarian menu yang dipilih oleh *customer* dan *counter* yang menyediakan menu tersebut, tabel yang digunakan adalah Tabel tb_trans_mstr, tb_trans_dtl, tb_food, dan tb_counter.

4.2.3.3 Data object description

Data object description menjelaskan secara rinci mengenai atribut-atribut yang dimiliki oleh masing-masing tabel yang ada pada basis data sesuai dengan *entity relationship diagram*. *Data object description* dapat dipisahkan menjadi dua bagian, yaitu :

1. *Data field description* menjelaskan keterangan setiap atribut pada masing- masing tabel yang ada di *database* food_court.
2. *Data type definition* menjelaskan tipe data yang digunakan oleh atribut pada masing-masing tabel yang ada di *database* food_court.

4.2.3.3.1 Data Field Description

Data field description menjelaskan keterangan seluruh kolom (*field*) dalam masing-masing tabel *database* food_court. *Data field description database* food_court ditunjukkan pada tabel-tabel dibawah ini:

Tabel 4.4 Data Field Description Tabel tb_customer

No	Field	Penjelasan
1	cust_id	Menunjukkan ID customer
2	cust_name	Menunjukkan nama customer
3	address	Menunjukkan alamat customer
4	phone	Menunjukkan nomer telepon customer
5	no_ktp	Menunjukkan nomer identitas customer
6	gender	Menunjukkan jenis kelamin customer
7	password	Menunjukkan password customer
8	email	Menunjukkan alamat email customer
9	credit_limit	Menunjukkan kredit yang dimiliki customer
10	delete_flag	Menunjukkan penghapusan data

11	create_by	Menunjukkan pembuat data
12	create_date	Menunjukkan kapan data dibuat

Sumber : Perancangan

Tabel 4.5 Data Field Description Tabel tb_counter

No	Field	Penjelasan
1	counter_id	Menunjukkan ID counter makanan
2	counter_name	Menunjukkan nama counter makanan
3	address	Menunjukkan alamat counter makanan
4	phone	Menunjukkan nomer telepon counter makanan
5	company_name	Menunjukkan nama perusahaan/ pemilik counter makanan
6	share_amt	Menunjukkan besarnya prosentase pembagian hasil
7	delete_flag	Menunjukkan penghapusan data
8	create_by	Menunjukkan pembuat data
9	create_date	Menunjukkan kapan data dibuat

Sumber : Perancangan

Tabel 4.6 Data Field Description Tabel tb_food

No	Field	Penjelasan
1	food_id	Menunjukkan ID makanan
2	counter_id	Menunjukkan ID counter yang menyediakan makanan tersebut
3	food_name	Menunjukkan nama makanan
4	price	Menunjukkan harga makanan
5	delete_flag	Menunjukkan penghapusan data
6	create_by	Menunjukkan pembuat data
7	create_date	Menunjukkan kapan data dibuat

Sumber : Perancangan

Tabel 4.7 Data Field Description Tabel tb_employee

No	Field	Penjelasan
1	emp_id	Menunjukkan ID employee / karyawan
2	emp_name	Menunjukkan nama karyawan
3	address	Menunjukkan alamat karyawan
4	position	Menunjukkan posisi/jabatan karyawan
5	gender	Menunjukkan jenis kelamin karyawan
6	email	Menunjukkan alamat email karyawan
7	counter_id	Menunjukkan ID counter dimana karyawan

		ditempatkan
8	delete_flag	Menunjukkan penghapusan data
9	create_by	Menunjukkan pembuat data
10	create_date	Menunjukkan kapan data dibuat

Sumber : Perancangan

Tabel 4.8 Data Field Description Tabel tb_trans_dtl

No	Field	Penjelasan
1	trans_id	Menunjukkan nomer transaksi
2	food_id	Menunjukkan ID makanan yang dipesan
3	qty	Menunjukkan banyaknya makanan yang dipesan
4	price	Menunjukkan harga makanan yang dipesan
5	disc	Menunjukkan besarnya diskon yang diberikan pada customer
6	status_order	Menunjukkan status dari order yang dilakukan oleh konsumen

Sumber : Perancangan

Tabel 4.9 Data Field Description Tabel tb_trans_hist

No	Field	Penjelasan
1	trans_date	Menunjukkan tanggal transaksi
2	trans_type	Menunjukkan jenis transaksi
3	counter_id	Menunjukkan ID counter makanan
4	customer_id	Menunjukkan ID customer
5	reff_no	Menunjukkan tanggal dilakukannya order
6	amount	Menunjukkan nama sales

Sumber : Perancangan

Tabel 4.10 Data Field Description Tabel tb_trans_mstr

No	Field	Penjelasan
1	emp_id	Menunjukkan ID karyawan
2	trans_id	Menunjukkan nomer transaksi
3	counter_id	Menunjukkan ID counter makanan
4	customer_id	Menunjukkan ID customer
5	date	Menunjukkan tanggal dilakukannya order
6	desk_no	Menunjukkan nomor meja
7	cancel_flag	Menunjukkan pembatalan order

8	create_by	Menunjukkan pembuat data
9	create_date	Menunjukkan kapan data dibuat

Sumber : Perancangan

Tabel 4.11 Data Field Description Tabel tb_user

No	Field	Penjelasan
1	user_id	Menunjukkan ID user
2	password	Menunjukkan password user
4	counter_id	Menunjukkan ID counter makanan
5	emp_id	Menunjukkan ID karyawan
8	delete_flag	Menunjukkan penghapusan data
9	create_by	Menunjukkan pembuat data
10	create_date	Menunjukkan kapan data dibuat

Sumber : Perancangan

4.2.3.3.2 Data Type Definition

Data type definition menjelaskan mengenai tipe data yang digunakan setiap tabel *database* food_court. Tabel-tabel yang terdapat pada basis data tersebut adalah :

- Tabel tb_customer merupakan entitas yang berfungsi untuk menyimpan data customer.

Tabel 4.12 Data type definition Tabel tb_customer

Field	Tipe	Panjang	Keterangan
cust_id	VARCHAR	10	NOT NULL, PRIMARY KEY
cust_name	VARCHAR	50	
address	VARCHAR	50	
phone	VARCHAR	50	
no_ktp	VARCHAR	50	
gender	CHAR	1	NOT NULL, DEFAULT "M" as male
password	CHAR	6	NOT NULL
email	VARCHAR	50	
credit_limit	CHAR	3	
delete_flag	VARCHAR	50	
create_by	DATETIME	-	

Sumber : Perancangan

- Tabel `tb_counter` merupakan entitas yang berfungsi untuk menyimpan data mengenai *counter* makanan.

Tabel 4.13 Data type definition Tabel `tb_counter`

Field	Tipe	Panjang	Keterangan
counter_id	VARCHAR	10	NOT NULL, PRIMARY KEY
counter_name	VARCHAR	50	
address	VARCHAR	50	
phone	VARCHAR	50	
company_name	VARCHAR	50	
share_amt	DOUBLE	-	
delete_flag	CHAR	1	
create_by	VARCHAR	50	
create_date	DATETIME	-	

Sumber :Perancangan

- Tabel `tb_food` merupakan entitas yang berfungsi untuk menyimpan data mengenai daftar menu yang disediakan oleh *counter* makanan.

Tabel 4.14 Data type definition Tabel `tb_food`

Field	Tipe	Panjang	Keterangan
food_id	VARCHAR	5	NOT NULL, PRIMARY KEY
counter_id	VARCHAR	10	FOREIGN KEY
food_name	VARCHAR	50	
price	DOUBLE	6	
delete_flag	CHAR	1	
create_by	VARCHAR	50	
create_date	DATETIME	-	

Sumber :Perancangan

- Tabel tb_employee merupakan entitas yang berfungsi untuk menyimpan data mengenai karyawan yang bekerja pada food court tersebut .

Tabel 4.15 Data type definition Tabel `tb_employee`

Field	Tipe	Panjang	Keterangan
emp_id	VARCHAR	5	NOT NULL, PRIMARY KEY
emp_name	VARCHAR	50	
address	VARCHAR	50	
position	VARCHAR	50	

gender	CHAR	1	NOT NULL, DEFAULT "M" as male
email	VARCHAR	50	
counter_id	VARCHAR	10	NOT NULL, PRIMARY KEY,FOREIGN KEY
delete_flag	CHAR	1	
create_by	VARCHAR	50	
create_date	DATETIME	-	

Sumber :Perancangan

- Tabel tb_trans_dtl merupakan entitas yang berfungsi untuk menyimpan data mengenai detail order makanan yang dilakukan oleh customer.

Tabel 4.16 Data type definition Tabel tb_trans_dtl

Field	Tipe	Panjang	Keterangan
trans_id	VARCHAR	10	NOT NULL, PRIMARY KEY
food_id	VARCHAR	5	NOT NULL, PRIMARY KEY, FOREIGN KEY
qty	VARCHAR	5	
price	INT	10	
disc	DOUBLE		
status_order	VARCHAR	10	

Sumber :Perancangan

- Tabel tb_trans_hist merupakan entitas yang berfungsi untuk menyimpan data mengenai transaksi yang dilakukan oleh customer.

Tabel 4.17 Data type definition Tabel tb_trans_hist

Field	Tipe	Panjang	Keterangan
trans_date	DATE TIME	5	NOT NULL, PRIMARY KEY
trans_type	VARCHAR	10	
counter_id	VARCHAR	10	NOT NULL, PRIMARY KEY, FOREIGN KEY
customer_id	VARCHAR	10	NOT NULL, PRIMARY KEY, FOREIGN KEY
reff_no	VARCHAR	3	
amount	DOUBLE	-	

Sumber :Perancangan

- Tabel tb_trans_mstr merupakan entitas yang berfungsi untuk menyimpan rekapan data transaksi customer yang diambil dari tabel tb_trans_dtl dan tb_trans_hist.

Tabel 4.18 Data type definition Tabel `tb_trans_mstr`

Field	Tipe	Panjang	Keterangan
emp_id	VARCHAR	10	NOT NULL, PRIMARY KEY, FOREIGN KEY
trans_id	INT UNSIGNED	5	NOT NULL, PRIMARY KEY
counter_id	VARCHAR	10	NOT NULL, PRIMARY KEY, FOREIGN KEY
customer_id	VARCHAR	10	NOT NULL, PRIMARY KEY, FOREIGN KEY
date	DATETIME	10	
desk_no	VARCHAR	3	
cancel_flag	CHAR	1	
create_by	VARCHAR	50	
create_date	DATETIME	-	

Sumber :Perancangan

- Tabel tb_user merupakan entitas yang berfungsi untuk menyimpan data mengenai user yang dapat mengakses Sistem Transaksi Pembayaran Food Court.

Tabel 4.19 Data type definition Tabel `tb_user`

Field	Tipe	Panjang	Keterangan
user_id	VARCHAR	10	NOT NULL, PRIMARY KEY
password	VARCHAR	10	NOT NULL
counter_id	VARCHAR	10	NOT NULL, PRIMARY KEY, FOREIGN KEY
emp_id	VARCHAR	10	NOT NULL, PRIMARY KEY, FOREIGN KEY
delete_flag	CHAR	1	
create_by	VARCHAR	50	
create_date	DATETIME	-	

Sumber :Perancangan

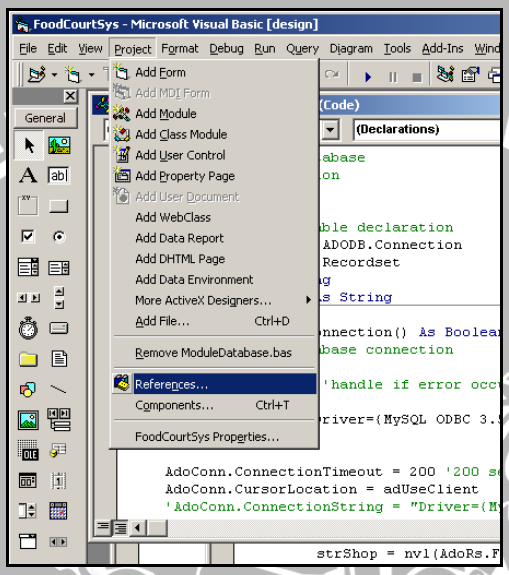
4.2.3.4 **Perancangan Koneksi *Database***

Sistem Transaksi Pembayaran *Food Court* menggunakan ADO *(Activex Data Object)* sebagai media untuk melakukan koneksi dengan *database* MySQL. ADO dapat diaplikasikan pada berbagai jenis *database* sehingga penggunaan ADO menjadi lebih fleksibel. ADO akan melakukan koneksi dengan ODBC pada sistem operasi *Microsoft Windows* melalui perangkat MyODBC. Perangkat lunak Sistem Transaksi Pembayaran Food Court dirancang untuk melakukan koneksi dengan ODBC tanpa menggunakan DSN *(Data Source Name)* sehingga perangkat

lunak dapat di*install* dengan mudah tanpa harus mengatur DSN pada setiap komputer. Koneksi ADO tanpa menggunakan DSN digantikan dengan mendefinisikan koneksi yang digunakan oleh ADO dengan menggunakan kode program. Untuk mendefinisikan koneksi tersebut, sebelumnya ADO akan didefinisikan sebagai *variabel* dan mereferensikan obyek ADO ke dalam *project.*

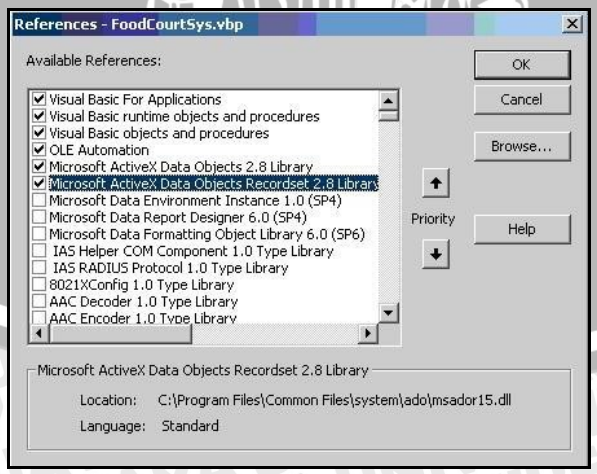
Hal tersebut dapat dilakukan dengan melakukan langkah-langkah:

1. Memilih menu *reference* pada Visual Basic seperti dalam Gambar 4.13.



Gambar 4.13 Menu *Reference*

2. Mereferensikan obyek ADO seperti dalam Gambar 4.14.



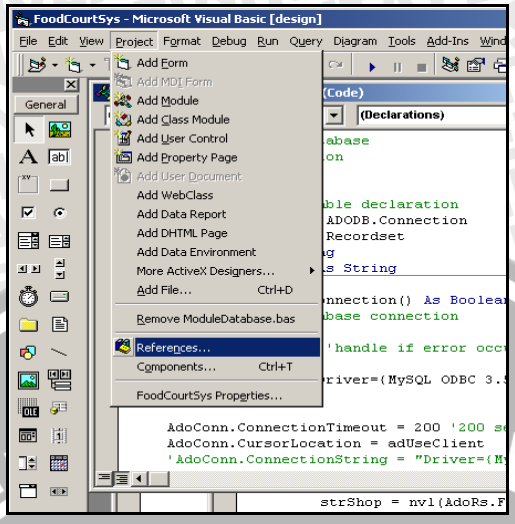
Gambar 4.14 Pilihan Referensi Obyek

3. Mendefinisikan koneksi ADO dengan *database* MySQL

```
Public AdoConn As New ADODB.Connection
Public AdoRs As ADODB.Recordset
Public strsql As String
Public strConnection As String

Public Function openConnection() As Boolean

    On Error GoTo err
    strConnection = "Driver={MySQL ODBC 3.51
    Driver};Server=localhost;Port=3306;Database=foodcourt;
    User=root; Password=prella;Option=3;"

    AdoConn.ConnectionTimeout = 200
    AdoConn.CursorLocation = adUseClient
    AdoConn.ConnectionString = strConnection
    AdoConn.Open
    If AdoConn.State = adStateOpen Then
              Set AdoRs = New ADODB.Recordset
        AdoRs.CursorLocation = adUseClient
    End If

    Call selectDatabase("foodcourt")
        openConnection = True


err:
    If err.Number <> 0 Then
MsgBox "Database connection error, " & err.Description,
vbCritical, "Warning"
        openConnection = False
    End If
```

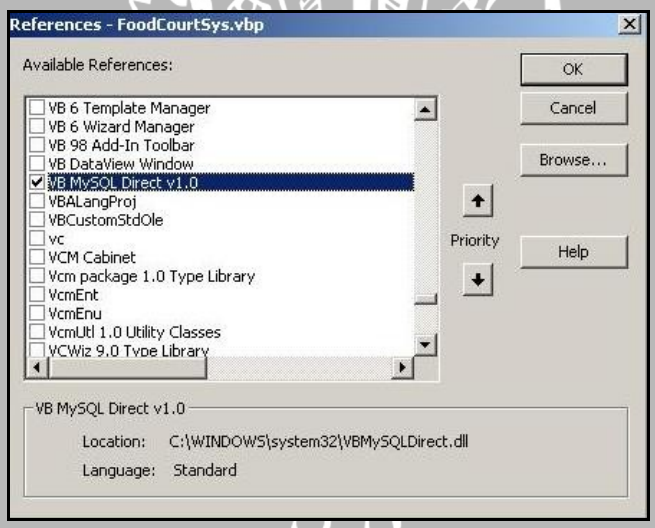
Selain menggunakan *MyODBC*, koneksi database dapat dilakukan dengan menggunakan *VBMySQLDirect*. *VBMySQLDirect* hanya digunakan untuk mengakses MySQL tetapi tidak dapat melakukan koneksi dengan SQL Server. VBMySQL memiliki kecepatan akses lebih tinggi dibandingkan penggunaan ADO+ODBC. Koneksi VBMySQLDirect didefinisikan dengan menggunakan kode program. Untuk mendefinisikan koneksi tersebut, sebelumnya akan didefinisikan sebagai *variabel* dan mereferensikan obyek VBMySQLDirect ke dalam *project.* Hal tersebut dapat dilakukan dengan melakukan langkah-langkah:

1. Memilih menu *reference* pada Visual Basic seperti dalam Gambar 4.15.



Gambar 4.15 Menu *Reference*

2. Mereferensikan obyek ADO seperti dalam Gambar 4.16.



Gambar 4.16 Pilihan Referensi Obyek

3. Mendefinisikan variable untuk koneksi dan recordset

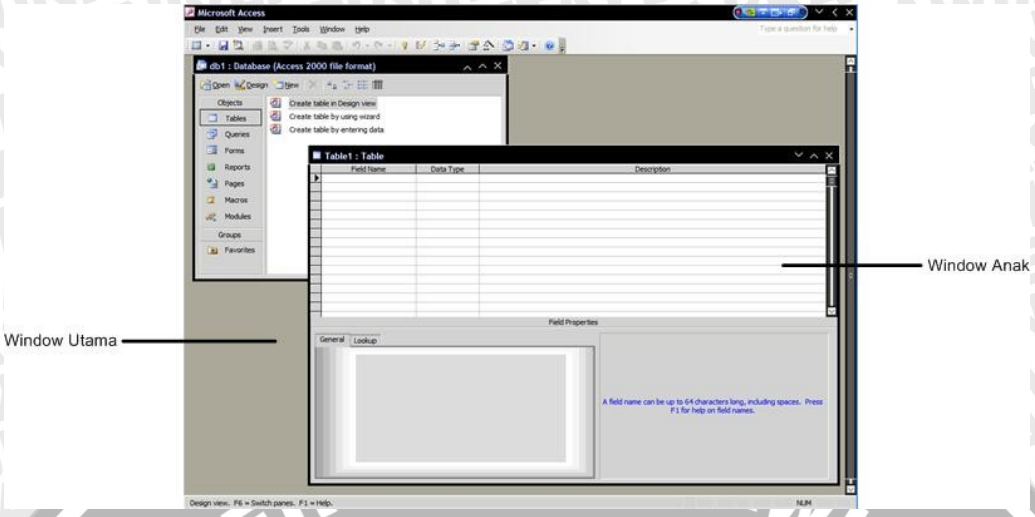
```
Public AdoConn As New MYSQL_CONNECTION
Public AdoRs As New MYSQL_RS
Public strsql As String
Public strConnection As String

Public Function openConnection() As Boolean

    On Error GoTo err
     strConnection
     ="SERVER=localhost;DATABASE=foodcourt;UID=root;PWD=p
     rella"

    AdoConn.ConnectionString = strConnection

     CallAdoConn.openConnection("localhost","root",
     "prella","foodcourt")

    If AdoConn.State = adStateOpen Then
         Set AdoRs = New MYSQL_RS
         AdoRs.CursorLocation = adUseClient
    End If

    Call selectDatabase("foodcourt")

    openConnection = True

err:
    If err.Number <> 0 Then
       MsgBox    "Database    connection    error,    "    &
       err.Description,        vbCritical,        "Warning"
       openConnection = False
    End If
```

4.2.4 Perancangan *User Interface*

Interface merupakan bagian yang menghubungkan antara sistem dengan *user*. Dengan demikian suatu perancangan *interface* yang baik akan memudahkan *user* untuk berinteraksi dengan sistem yang dibuat.

Interface perangkat lunak Sistem Transaksi Pembayaran *Foodcourt* menggunakan jenis MDI (*Multiple Document Interface*). Keuntungan penggunaan MDI adalah form-form yang akan ditampilkan pada Sistem Transaksi Pembayaran *Food court* dapat diorganisir dengan baik, sehingga pengguna/*user* aplikasi tidak mengalami kesulitan dalam menggunakan sistem.

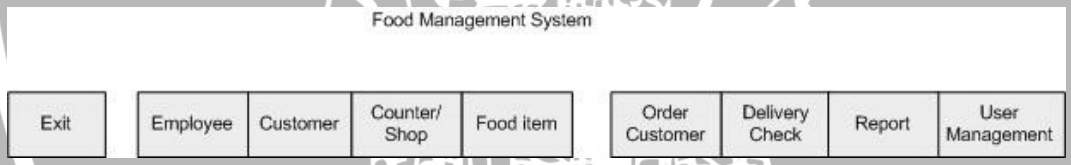
Interface MDI memiliki dua bagian yaitu window utama yang disebut MDI form dan window anak. Window utama akan menjadi tempat untuk menampilkan form anak. Jenis *interface* MDI yang telah diaplikasikan pada aplikasi sistem operasi Microsoft Windows diperlihatkan pada Gambar 4.17:



Gambar 4.17 Interface MDI pada Microsoft Access

4.2.4.1 Tampilan halaman utama

Halaman utama dari sistem akan memuat beberapa menu, yaitu : *exit,employee, customer, counter, food item, order customer, Delivery Check*, *report* dan *user management*. Adapun rancangan dari halaman utama diperlihatkan pada gambar 4.18:



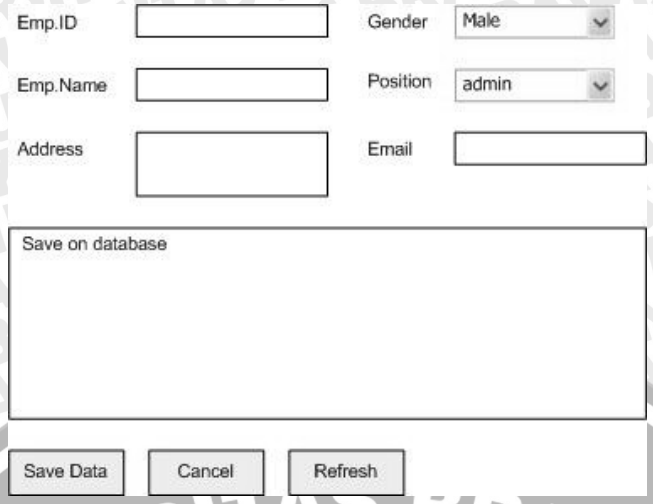
Gambar 4.18 Halaman Utama Sistem Transaksi Pembayaran Food Court

Sumber :Perancangan

4.2.4.2 Tampilan Halaman Employee

Pada halaman *Employee* terdapat tiga sub menu, yaitu *Employee registration, Edit Employee*, dan *Report*. Halaman Employee hanya dapat diakses oleh admin aplikasi.

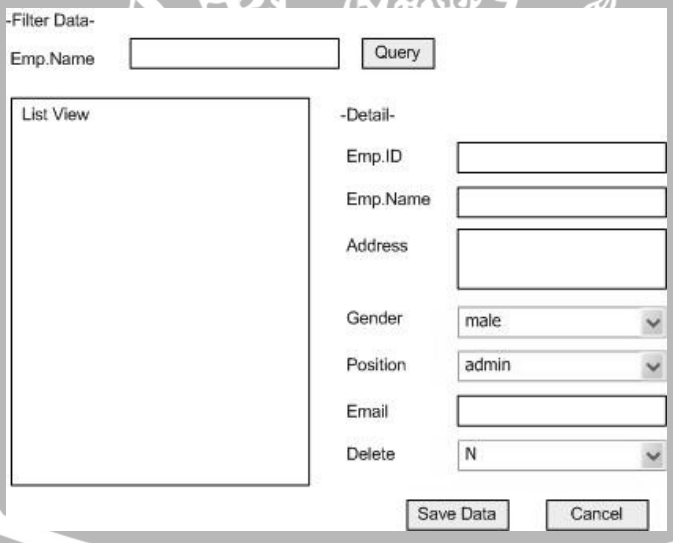
- Form *Employee registration* digunakan untuk memuat data mengenai karyawan *food court*, pada form ini terdapat menu *save data* untuk menyimpan data, *cancel* untuk membatalkan data yang akan disimpan, dan *refresh*. Adapun rancangan dari form tersebut dapat dilihat pada Gambar 4.19:



Gambar 4.19 Form *Employee registration*

Sumber : Perancangan

- Form *Edit Employee* digunakan apabila terdapat perubahan data mengenai karyawan *food court*, pada form ini terdapat menu *query* untuk mencari data karyawan berdasarkan nama, *save data* untuk menyimpan perubahan data, dan *cancel* untuk membatalkan data yang akan disimpan. Rancangan dari form tersebut diperlihatkan pada Gambar 4.20:

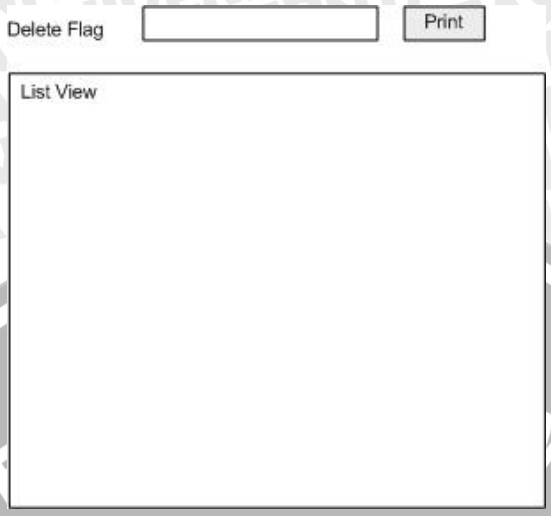


Gambar 4.20 Form *Edit Employee*

Sumber : Perancangan

- Form *Report* digunakan untuk merekap keseluruhan data karyawan *foodcourt*, pada form ini terdapat menu *print* yang digunakan untuk mencetak data yang diperlukan. Form *report* ini nantinya juga digunakan

pada halaman lainnya, yaitu halaman untuk *customer*, *counter/shop*, dan *food item*. Rancangan dari form *report* diperlihatkan pada Gambar 4.21:



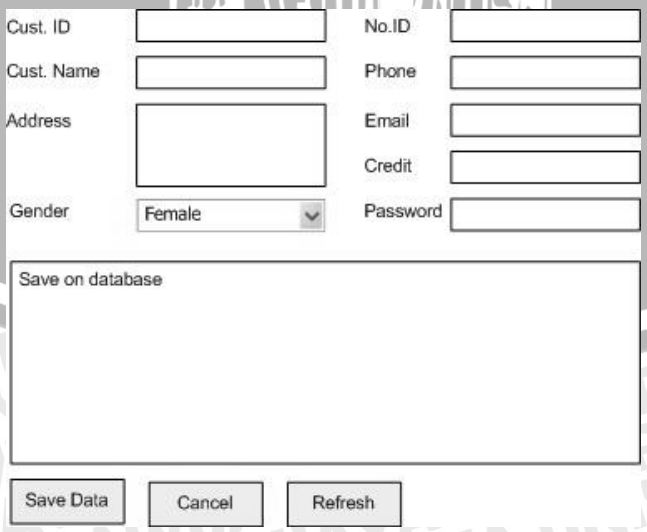
Gambar 4.21 Form *Report*

Sumber : Perancangan

4.2.4.3 Tampilan Halaman Customer

Pada halaman Customer terdapat tiga sub menu, yaitu *Customer registration, Edit Customer*, dan *Report.*

- Form *customer registration* digunakan untuk memuat data mengenai *customer* yang terdaftar sebagai anggota, pada form ini terdapat menu *save data* untuk menyimpan data, *cancel* untuk membatalkan data yang akan disimpan, dan *refresh*. Adapun rancangan dari form tersebut dapat dilihat pada gambar 4.22 :



Gambar 4.22 Form *customer registration*

Sumber : Perancangan

- Form *Edit customer* digunakan apabila terdapat perubahan data mengenai *customer*, pada form ini terdapat menu *query* untuk mencari data customer berdasarkan nama, *save data* untuk menyimpan perubahan data, dan *cancel* untuk membatalkan data yang akan disimpan. Rancangan dari form tersebut diperlihatkan pada Gambar 4.23:

Gambar 4.23 Form *Edit customer*

Sumber : Perancangan

- Form *Report* digunakan untuk merekap keseluruhan data *customer*, pada form ini terdapat menu *print* yang digunakan untuk mencetak data yang diperlukan. Rancangan dari form *report* telah diperlihatkan pada Gambar 4.21.

4.2.4.4 Tampilan Halaman Counter/shop

Pada halaman counter/shop terdapat tiga sub menu, yaitu *Counter/Shop registration*, *Edit Counter/Shop*, dan *Report*.

- Form *counter/shop registration* digunakan untuk memuat data mengenai *counter/shop* yang terdapat pada *foodcourt*, pada form ini terdapat menu *save data* untuk menyimpan data, *cancel* untuk membatalkan data yang akan disimpan, dan *refresh*. Adapun rancangan dari form tersebut dapat dilihat pada gambar 4.24 :

Gambar 4.24 Form *counter/shop registration*

Sumber : Perancangan

- Form *Edit counter/shop* digunakan apabila terdapat perubahan data mengenai *counter/shop*, pada form ini terdapat menu *query* untuk mencari data *counter* berdasarkan nama *counter*, *save data* untuk menyimpan perubahan data, dan *cancel* untuk membatalkan data yang akan disimpan. Rancangan dari form tersebut diperlihatkan dalam Gambar 4.25:

Gambar 4.25 Form *Edit counter/shop*

Sumber : Perancangan

- Form *Report* digunakan untuk merekap keseluruhan data *counter*, pada form ini terdapat menu *print* yang digunakan untuk mencetak data yang

diperlukan. Rancangan dari form *report* telah diperlihatkan dalam Gambar 4.21.

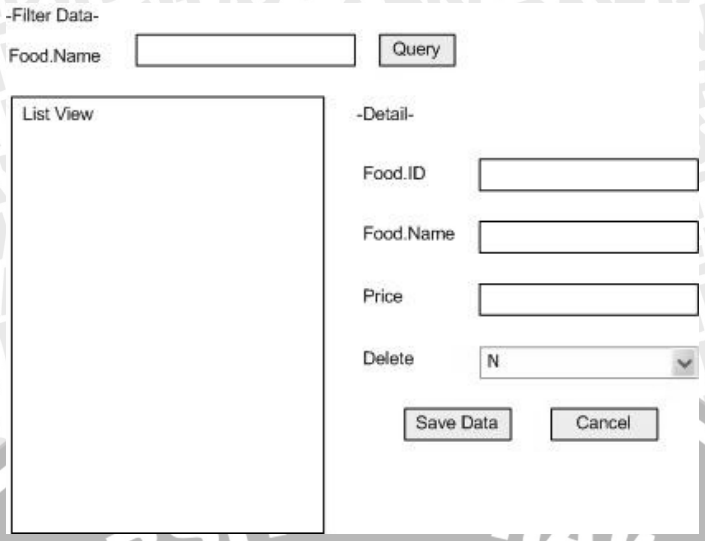
4.2.4.5 Tampilan Halaman Food/Item

Pada halaman *Food/Item* terdapat tiga sub menu, yaitu *Food/Item registration, Edit Food/Item*, dan *Report*.

- Form *Food/Item registration* akan untuk memuat data mengenai *Food/Item* yang tersedia pada tiap *counter*. Pada form ini terdapat menu *save data* untuk menyimpan data, *cancel* untuk membatalkan data yang akan disimpan, dan *refresh*. Adapun rancangan dari form tersebut dapat dilihat pada gambar 4.26 :

Gambar 4.26 Form *Food/Item registration*
Sumber : Perancangan

- Form *Edit food/item* digunakan apabila terdapat perubahan data mengenai *food/item* yang disediakan oleh *counter*, pada form ini terdapat menu *query* untuk mencari data makanan berdasarkan nama makanan, *save data* untuk menyimpan perubahan data, dan *cancel* untuk membatalkan data yang akan disimpan. Rancangan dari form tersebut diperlihatkan pada Gambar 4.27:



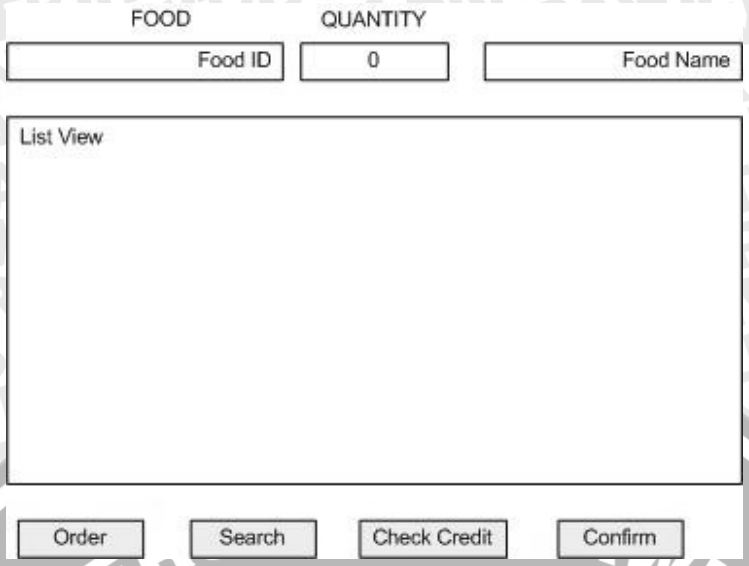
Gambar 4.27 Form *Edit food/item*

Sumber : Perancangan

- Form *Report* digunakan untuk merekap keseluruhan data *counter*, pada form ini terdapat menu *print* yang digunakan untuk mencetak data yang diperlukan. Rancangan dari form report telah diperlihatkan dalam Gambar 4.21.

4.2.4.6 Tampilan Halaman *Order Customer*

Pada halaman *order customer* memuat daftar makanan yang di pesan oleh *customer*. Pada form ini terdapat menu *order* untuk memilih makanan yang dipesan, menu *search* untuk membantu operator *counter* mengetahui ID makanan yang dipesan, menu *check credit* untuk mengetahui sisa kredit yang terdapat pada *E-card*, dan menu *confirm* untuk mengkonfirmasi pesanan customer dengan memindai *customer* ID pada *E-card* dan mengetahui nomor meja yang dipilih *customer*. Halaman *order customer* hanya diakses oleh operator *counter*. Adapun rancangan dari form tersebut dapat dilihat pada gambar 4.28 :

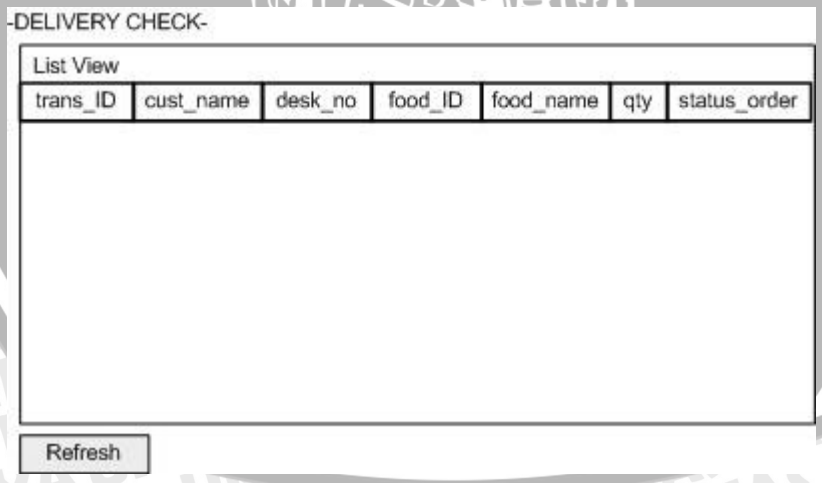


Gambar 4.28 Halaman *Order Customer*

Sumber : Perancangan

4.2.4.7 Tampilan Halaman *Delivery Check*

Halaman *Delivery Check* memuat list order *customer* yang harus segera diantar, status order akan diubah sesaat sebelum pelayan mengantarkan pesanan. Halaman *Delivery Check* memiliki menu *refresh*, dimana setiap order yang berstatus '*deliver'* akan terhapus secara otomatis dari list order. Rancangan dari form tersebut diperlihatkan pada Gambar 4.29:



Gambar 4.29 Halaman *Delivery Check*

Sumber : Perancangan

4.2.4.8 Tampilan Halaman Report

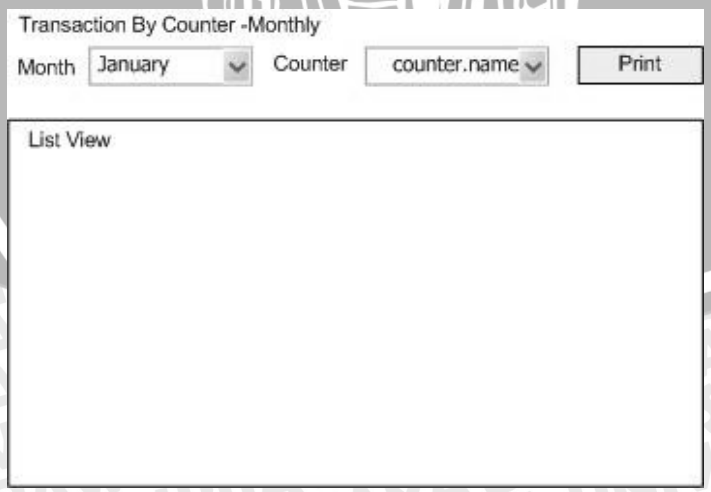
Pada halaman report terbagi atas 4 sub menu, yaitu : *transaction by counter-daily*, *transaction by counter-monthly*, *share by shop,* dan *customer transaction*.

- Form *transaction by counter-daily* memuat list transaksi yang terjadi pada setiap *counter* makanan dalam sehari, sekaligus besarnya pendapatan yang diperoleh setiap *counter*. Rancangan form tersebut diperlihatkan pada Gambar 4.30:

Gambar 4.30 Form *transaction by counter-daily*

Sumber : Perancangan

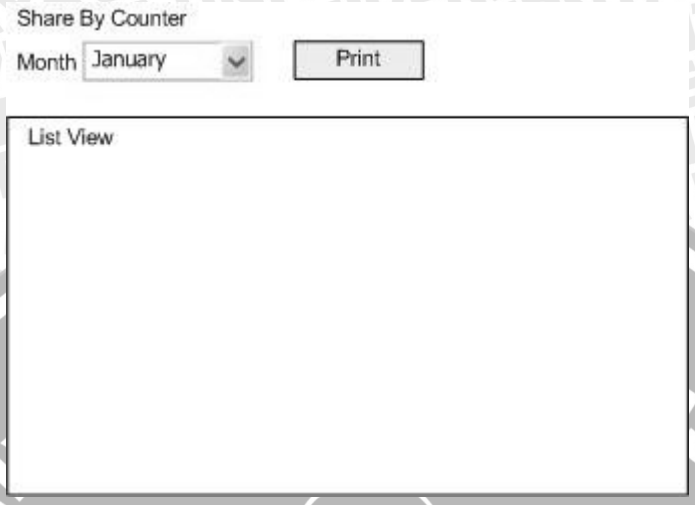
- Form *transaction by counter-monthly* memuat list transaksi yang terjadi pada setiap *counter* makanan dalam satu bulan, sekaligus besarnya pendapatan yang diperoleh setiap *counter*. Rancangan form tersebut diperlihatkan pada Gambar 4.31:



Gambar 4.31 Form *transaction by counter-monthly*

Sumber : Perancangan

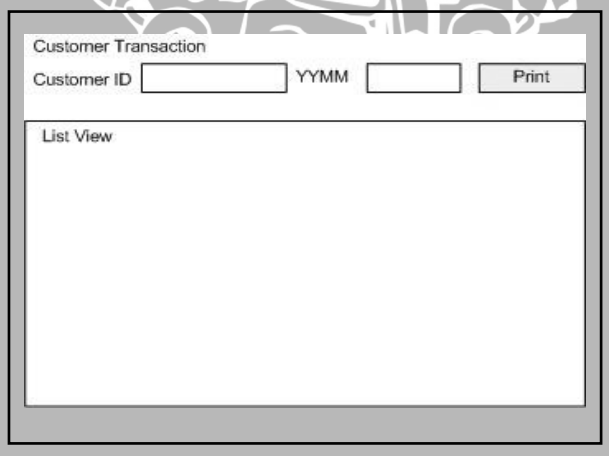
- Form *share by shop* memuat laporan mengenai pembagian hasil dari transaksi setiap *counter* makanan dalam selang waktu satu bulan. Rancangan form tersebut diperlihatkan pada Gambar 4.32:



Gambar 4.32 Form *share by shop*

Sumber : Perancangan

- Form customer transaction memuat laporan mengenai transaksi yang telah dilakukan oleh customer dalam selang waktu satu bulan. Rancangan form tersebut diperlihatkan pada Gambar 4.33:



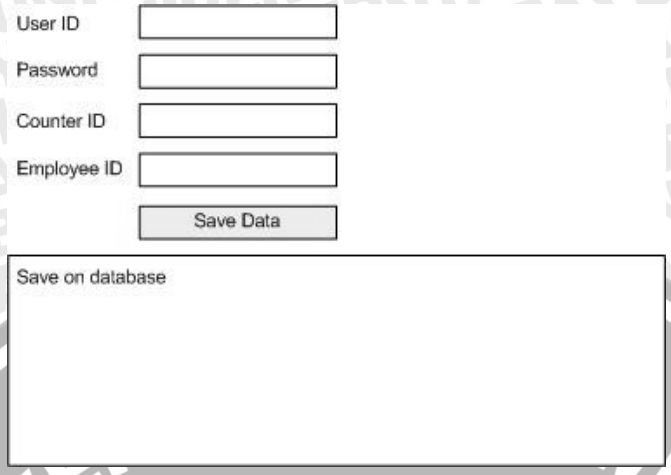
Gambar 4.33 Form *customer transaction*

Sumber : Perancangan

4.2.4.9 Tampilan Halaman User Management

Form User Management digunakan untuk memuat data mengenai user yang berhak untuk mengakses Sistem Transaksi Pembayaran Food Court, pada form ini terdapat menu *save data* untuk menyimpan data *user*, *cancel* untuk

membatalkan data *user* yang akan disimpan, dan *refresh*. Adapun rancangan dari form tersebut dapat dilihat pada Gambar 4.34 :



Gambar 4.34 Form *user management*

Sumber : Perancangan