

LAPORAN PRAKTIKUM BASIS DATA
**“Pembuatan Database Kasir Rawat Inap dan
Pengimplementasiannya Dalam Bentuk Website”**
Disusun Untuk Memenuhi Tugas Besar Praktikum Basis Data
Dosen Pengampu : Ridwan Setiawan, S.T., M.Kom.



Disusun oleh :
Habil Gymnastiar Abdul Matin – 2106008
Sahrudin Fiqri Muzahidat - 2106024
Teknik Informatika B

PROGRAM STUDI TEKNIK INFORMATIKA
INSTITUT TEKNOLOGI GARUT

2023

KATA PENGANTAR

Puji syukur saya panjatkan kehadiran Allah SWT yang telah menganugerahkan banyak nikmat sehingga saya dapat menyusun laporan praktikum Basis Data yang berjudul **“Pembuatan Database Kasir Rawat Inap dan Pengimplementasiannya Dalam Bentuk Website”** ini dan dapat terselesaikan tepat pada waktunya. Penulisan laporan ini bertujuan untuk menambah wawasan bagi penulis atau pembaca mengenai **“Pembuatan Database Kasir Rawat Inap dan Pengimplementasiannya Dalam Bentuk Website”** serta untuk memenuhi tugas besar Praktikum Basis Data.

Dalam penyusunan laporan ini, kami menyadari bahwa hasil laporan praktikum ini masih jauh dari kata sempurna. Sehingga kami selaku penyusun sangat mengharapkan kritik dan saran yang membangun dari pembaca sekalian. Akhir kata, semoga laporan praktikum ini dapat memberikan manfaat bagi penyusun maupun bagi pembaca.

Penyusun

Habil Gymnastiar Abdul Matin

DAFTAR ISI

KATA PENGANTAR	2
DAFTAR ISI	3
DAFTAR GAMBAR	4
DAFTAR TABEL	5
BAB I Pendahuluan.....	6
1.1 Latar Belakang.....	6
1.2 Rumusan Masalah	7
1.3 Tujuan.....	7
BAB II TINJAUAN PUSTAKA.....	8
2.1 ERD (Entity Relationship Diagram)	8
2.2 Model Relasi Entitas (Entity-Relationship Model)	8
2.3 Cardinality (Kardinalitas) Relasi	8
2.4 Atribut.....	9
BAB III HASIL DAN PEMBAHASAN.....	11
3.1 Studi Kasus	11
3.2 Tahapan Pembuatan ERD.....	11
3.3 Membuat Database pada mySQL.....	13
3.4 Tahapan Pembuatan DDL	14
3.5 Data Manipulation Language	18
3.6 Implementasi CRUD	20
BAB IV KESIMPULAN.....	33
DAFTAR PUSTAKA	34
LAMPIRAN	35

DAFTAR GAMBAR

Gambar 3. 1 Diagram ERD	12
Gambar 3. 2 Tampilan XAMPP	13
Gambar 3. 3 Perintah awal pada XAMPP	13
Gambar 3. 4 Menampilkan database	14
Gambar 3. 5 Membuat database baru	14
Gambar 3. 6 Menampilkan database baru	14
Gambar 3. 7 Mengaktifkan database	15
Gambar 3. 8 Membuat tabel baru	15
Gambar 3. 9 Menambahkan tabel baru yaitu tabel kamar	15
Gambar 3. 10 Menambahkan tabel baru yaitu tabel administrasi	15
Gambar 3. 11 Menampilkan nama tabel	16
Gambar 3. 12 Menampilkan isi pada tabel	16
Gambar 3. 13 Merubah dan memodifikasi kolom	16
Gambar 3. 14 Menambah kolom pada tabel	17
Gambar 3. 15 Menghapus kolom	17
Gambar 3. 16 Rename tabel	17
Gambar 3. 17 Menghapus tabel	18
Gambar 3. 18 Menghapus databse	18
Gambar 3. 19 Perintah insert	19
Gambar 3. 20 Perintah select	19
Gambar 3. 21 Perintah update	20
Gambar 3. 22 Perintah delete	20
Gambar 3. 23 Implementasi Create	21
Gambar 3. 24 Implementasi Create	21
Gambar 3. 25 Implementasi Read	21
Gambar 3. 26 Implementasi Update	22
Gambar 3. 27 Implementasi Delete	22

DAFTAR TABEL

Tabel 3. 1 Isi Entitas, Atribut, dan Relasi untuk membuat ERD	11
Tabel 3. 2 Menentukan relasi antar entitas	12
Tabel 3. 3 Menentukan Cardinalitas antar entitas.....	12

BAB I

Pendahuluan

1.1 Latar Belakang

Dalam dunia bisnis, pengelolaan data menjadi hal yang sangat penting dalam menjalankan kegiatan operasional. Terutama dalam industri perhotelan atau rumah sakit, pengelolaan data transaksi keuangan merupakan aspek kritis yang harus diperhatikan dengan baik. Salah satu bagian penting dalam pengelolaan keuangan adalah sistem kasir yang digunakan untuk mencatat transaksi penjualan dan pembayaran.

Dalam rangka meningkatkan efisiensi dan akurasi pengelolaan keuangan, penggunaan database dalam sistem kasir telah menjadi solusi yang umum digunakan. Database kasir yang baik memungkinkan pengguna untuk mencatat, mengelola, dan melacak transaksi keuangan dengan lebih mudah dan efisien. Selain itu, dengan mengimplementasikan database kasir dalam bentuk website, informasi keuangan dapat diakses secara online, sehingga memudahkan manajemen untuk memantau dan mengontrol arus kas dari jarak jauh.

Dalam laporan praktikum ini, dilakukan pembuatan database kasir rawat inap dan pengimplementasiannya dalam bentuk website. Tujuan dari laporan ini adalah untuk menyajikan sebuah solusi yang terintegrasi dan efisien dalam pengelolaan keuangan di sektor rawat inap. Dalam hal ini, sistem kasir rawat inap adalah sistem yang digunakan untuk mencatat transaksi pembayaran dari pasien atau tamu yang menggunakan layanan rawat inap di rumah sakit atau klinik.

Pembuatan database kasir rawat inap akan melibatkan perancangan struktur database yang tepat, termasuk tabel-tabel yang diperlukan untuk menyimpan informasi tentang pasien, kamar rawat inap, jenis layanan, pembayaran, dan lain sebagainya. Selain itu, perancangan antarmuka website juga akan menjadi bagian penting dalam implementasi sistem kasir ini. Antarmuka website akan memberikan aksesibilitas yang mudah bagi pengguna untuk memasukkan dan mengambil informasi transaksi keuangan.

Penerapan database kasir rawat inap dalam bentuk website memiliki beberapa keuntungan. Pertama, memungkinkan pengguna untuk mengakses informasi keuangan secara real-time, sehingga manajemen dapat dengan cepat memantau arus kas dan mengambil tindakan yang diperlukan. Kedua, penggunaan website sebagai antarmuka memudahkan interaksi dengan sistem kasir, tanpa perlu menginstal aplikasi khusus di setiap perangkat pengguna. Ketiga, adanya database kasir akan membantu mengurangi kesalahan manusia dalam mencatat transaksi, sehingga meningkatkan akurasi data keuangan.

Dengan adanya laporan praktikum ini, diharapkan dapat memberikan pemahaman yang lebih baik tentang pentingnya penggunaan

database kasir dalam pengelolaan keuangan di sektor rawat inap. Selain itu, laporan ini juga diharapkan dapat memberikan panduan praktis tentang pembuatan database kasir rawat inap dan implementasinya dalam bentuk website. Dengan demikian, diharapkan pengguna dapat mengoptimalkan pengelolaan keuangan mereka, meningkatkan efisiensi operasional, dan memberikan pelayanan yang lebih baik bagi pasien atau tamu yang menggunakan layanan rawat inap.

1.2 Rumusan Masalah

Rumusan masalah yang akan dibahas:

1. Bagaimana proses pembuatan ERD?
2. Bagaimana proses pembuatan Database?
3. Bagaimana cara pengimplementasian database dalam bentuk Tampilan website?

1.3 Tujuan

Adapun tujuan dari laporan praktikum kali ini sebagai berikut:

1. Mengetahui proses pembuatan ERD.
2. Mengetahui proses pembuatan database.
3. Mengetahui cara mengimplementasikan database dalam bentuk Tampilan website.
4. Membantu pengelola dalam mengelola data kos-kosan yang ada.

BAB II

TINJAUAN PUSTAKA

2.1 ERD (Entity Relationship Diagram)

ERD memiliki beberapa aliran notasi seperti notasi Chen (dikembangkan oleh Peter Chen), Barker (dikembangkan oleh Richard Barker, Ian Palmer, Harry Ellis), notasi Crow's Foot, dan beberapa notasi lain. Namun yang banyak digunakan adalah notasi dari Chen. Entity Relationship Diagram (ERD) merupakan gambaran yang merelasikan antara objek yang satu dengan objek yang lain dari objek didunia nyata yang sering dikenal dengan hubungan antar entitas. Berikut adalah simbol-simbol yang digunakan pada ERD dengan notasi Chen:

1. Entitas : suatu yang nyata atau abstrak yang mempunyai karakteristik dimana kita akan menyimpan data.
2. Atribut : ciri umum semua atau sebagian besar instansi pada entitas tertentu.
3. Relasi : hubungan alamiah yang terjadi antara satu atau lebih entitas.
4. Link : garis penghubung atribut dengan kumpulan entitas dan kumpulan entitas dengan relasi.

2.2 Model Relasi Entitas (Entity-Relationship Model)

Model Relasi Entitas (Entity-Relationship Model) adalah sebuah model data konseptual yang digunakan untuk memodelkan hubungan antara objek atau entitas dalam sebuah basis data. Model ini dikembangkan oleh Peter Chen pada tahun 1976, dan menjadi salah satu model data paling umum yang digunakan dalam pengembangan basis data.

Model Relasi Entitas pada hakekatnya perwujudan dari model relasional dalam bentuk diagram, yaitu ER Diagram. Model Relasi Entitas terdiri dari tiga elemen dasar yaitu entitas, atribut, dan hubungan. Entitas mewakili objek atau konsep yang ingin diatur dalam basis data, seperti pelanggan, produk, atau pesanan. Atribut adalah karakteristik dari entitas, seperti nama pelanggan, alamat, atau nomor telepon. Hubungan menggambarkan keterkaitan antara entitas, seperti hubungan antara pelanggan dan pesanan.

2.3 Cardinality (Kardinalitas) Relasi

Cardinality Ratio Constraint atau batasan rasio kardinalitas adalah salah satu jenis constraint atau batasan dalam desain basis data yang mengontrol hubungan antara entitas atau tabel dalam database. Constraint ini

membatasi rasio antara jumlah baris atau rekord dalam satu tabel dengan jumlah baris dalam tabel yang terkait.

Jenis-jenis Cardinality Ratio Constraint:

1. One-to-One (1:1)
Batasan ini menunjukkan bahwa satu baris pada tabel pertama hanya berhubungan dengan satu baris pada tabel kedua, dan sebaliknya. Contoh: hubungan antara data KTP dengan data SIM.
2. One-to-Many (1:M)
Batasan ini menunjukkan bahwa satu baris pada tabel pertama bisa berhubungan dengan banyak baris pada tabel kedua, tetapi setiap baris pada tabel kedua hanya dapat berhubungan dengan satu baris pada tabel pertama. Contoh: hubungan antara data kependudukan dengan data keluarga.
3. Many-to-One (M:1)
Batasan ini menunjukkan bahwa banyak baris pada tabel pertama bisa berhubungan dengan satu baris pada tabel kedua, tetapi setiap baris pada tabel pertama hanya dapat berhubungan dengan satu baris pada tabel kedua. Contoh: hubungan antara data keluarga dengan data kependudukan.
4. Many-to-Many (M;M)
Batasan ini menunjukkan bahwa banyak baris pada tabel pertama bisa berhubungan dengan banyak baris pada tabel kedua. Contoh: hubungan antara data kependudukan dengan data pendidikan.

2.4 Atribut

Atribut dalam entitas adalah ciri atau informasi yang menjelaskan sifat dari suatu entitas atau objek dalam sebuah database. Dalam konsep entitas, atribut merupakan karakteristik atau sifat dari suatu entitas yang harus diidentifikasi, diukur, dan disimpan di dalam database.

Jenis-jenis atribut dalam entitas adalah sebagai berikut:

- 1) Atribut Sederhana (Simple Attribute)
Atribut yang hanya memiliki satu nilai atau nilai tunggal. Contohnya: nama, alamat, tanggal lahir, dan nomor telepon.
- 2) Atribut Komposit (Composite Attribute)
Atribut yang terdiri dari beberapa atribut sederhana. Contohnya: alamat yang terdiri dari atribut jalan, kota, provinsi, dan kode pos.
- 3) Atribut Turunan (Derived Attribute)
Atribut yang nilainya diperoleh dari hasil perhitungan atau operasi matematika atau logika dari atribut-atribut lain pada

entitas 5 yang sama. Contohnya: usia, yang dihitung berdasarkan tanggal lahir.

4) Atribut Multivalued (Multi-Valued Attribute)

Atribut yang memiliki lebih dari satu nilai. Contohnya: hobi, yang bisa memiliki beberapa nilai seperti berenang, membaca, dan menulis.

5) Atribut Kunci (Key Attribute)

Atribut yang digunakan untuk mengidentifikasi suatu entitas secara unik. Contohnya: nomor identitas, nomor KTP, atau nomor induk mahasiswa. Terdapat beberapa jenis atribut kunci (key) dalam pengolahan data dan basis data, di antaranya adalah:

- a) Primary Key, Atribut kunci utama yang unik dan dapat mengidentifikasi setiap baris (row) dalam sebuah tabel. Primary key biasanya terdiri dari satu atau beberapa kolom pada tabel.
- b) Foreign Key, Atribut yang mengacu pada primary key di tabel lain, digunakan untuk menjalin hubungan antar tabel dalam sebuah basis data.
- c) Candidate Key, Atribut atau kombinasi atribut yang dapat digunakan sebagai primary key. Setiap candidate key harus unik dan dapat mengidentifikasi setiap baris dalam tabel.
- d) Alternate Key, Atribut atau kombinasi atribut yang dapat digunakan sebagai primary key jika primary key yang ada tidak dapat digunakan.
- e) Composite Key, yaitu Primary key yang terdiri dari dua atau lebih atribut.
- f) Surrogate Key, yaitu Primary key buatan yang ditambahkan ke sebuah tabel, biasanya berupa angka atau kode unik, digunakan ketika tidak ada atribut yang cocok untuk menjadi primary key.

BAB III

HASIL DAN PEMBAHASAN

3.1 Studi Kasus

Dalam sebuah rumah sakit bernama “Rumah Sakit Guntur”, terdapat kebutuhan untuk mengimplementasikan sistem kasir rawat inap yang efisien dan terintegrasi. Sistem ini dirancang untuk mencatat dan mengelola transaksi keuangan yang terkait dengan pasien yang menggunakan layanan rawat inap di rumah sakit tersebut. Entitas utama yang terlibat dalam sistem kasir ini adalah pasien, administrasi, dan kamar.

Tabel 3. 1 Isi Entitas, Atribut, dan Relasi untuk membuat ERD

Entitas	Pasien	Administrasi	Kamar
Atribut	1. ID_pasien 2. Nama_pasien 3. Usia 4. Jenis_kelamin 5. Alamat	1. ID_adm 2. ID_pasien 3. No_kamar 4. Nama_adm 5. Tagihan	1. No_kamar 2. Nama_kamar 3. Jumlah_kamar
Relasi	Mendata, memilih		

3.2 Tahapan Pembuatan ERD

Berikut ini adalah tahapan-tahapan dalam pembuatan ERD nya:

1. Penentuan Entitas

Penentuan entitas dilakukan untuk mendapatkan hasil identifikasi yang ada dalam lingkungan pemakai. Entitas dapat merupakan sebuah elemen lingkungan dari perusahaan , seperti customer atau supplier suatu sumber daya , seperti suatu piutang dagang, suatu produk, atau suatu penjual suatu arus informasi , seperti suatu penjualan, pemesanan atau suatu faktur Di dalam sebuah entitas terdapat beberapa atribut.

Berikut ini adalah entitas dari yang telah kami pilih untuk studi kasus sistem kasir rawat inap.

- i. Entitas Pasien
- ii. Entitas Administrasi
- iii. Entitas Kamar

2. Menentukan Atribut Entitas

Setelah tahap menentukan entitas, berikut nya adalah menentukan atribut masing-masing entitas tersebut. Penentuan atribut dilakukan untuk mendeskripsikan karakter entitas. Berikut atributnya:

1. Pasien : - *id_pasien (int) PrimaryKey

- nama_pasien (varchar(25))
- usia (varchar(3))
- jenis_kelamin (varchar(9))
- alamat (varchar(30))
- 2. Administrasi :
 - *id_adm (int) PrimaryKey
 - **id_pasien (int) ForeignKey
 - **no_kamar (int) ForeignKey
 - nama_adm (varchar(10))
 - tagihan (num)
- 3. Kamar :
 - *no_kamar (int) PrimaryKey
 - nama_kamar (varchar(10))
 - jumlah_kamar (int)

3. Menentukan Relasi dan Cardinalitas

Tabel 3. 2 Menentukan relasi antar entitas

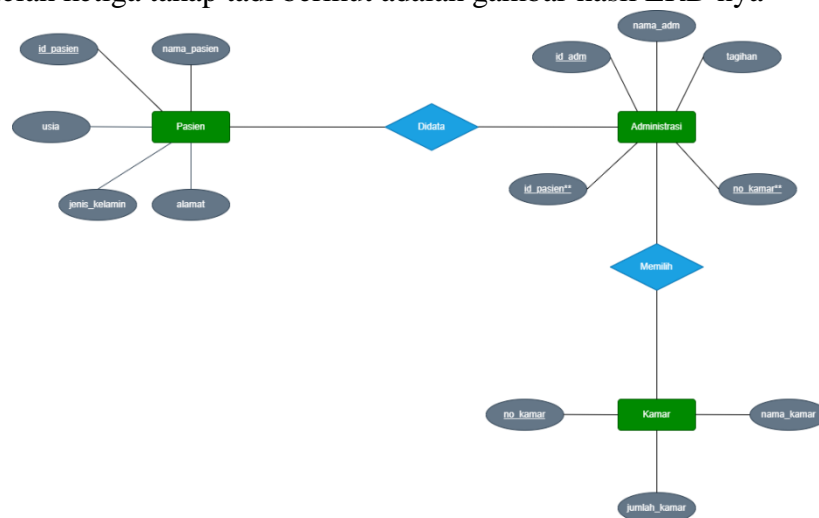
	Pasien	Administrasi	Kamar
Pasien	-	didata	-
Administrasi	mendata	-	Memilih
Kamar	-	dipilih	-

Tabel 3. 3 Menentukan Cardinalitas antar entitas

	Pasien	Administrasi	Kamar
Pasien	-	m:1	-
Administrasi	1:m	-	1:1
Kamar	-	1:1	-

4. Pembuatan Diagram ERD

Setelah ketiga tahap tadi berikut adalah gambar hasil ERD nya

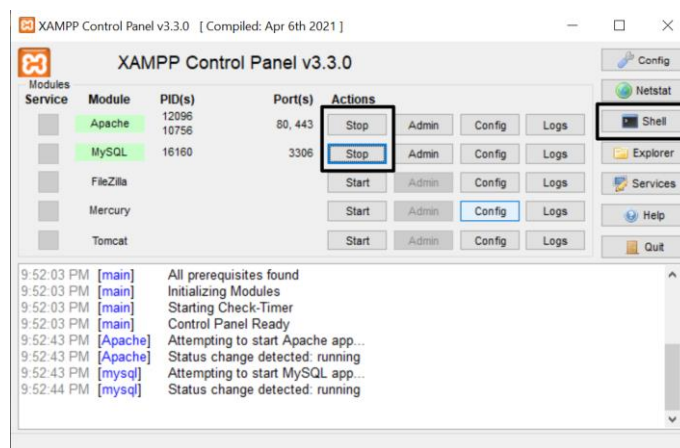


Gambar 3. 1 Diagram ERD

3.3 Membuat Database pada mySQL

Pada pertemuan pertama kita sudah belajar cara menginstal aplikasi XAMPP dan MySQL, pada kali ini kita akan membuat database menggunakan aplikasi Xamp dengan menggunakan MySQL. Adapun data yang akan di masukan adalah daa yang samásperti di atas megenai kemasyarakatan. Berikut ini tahap pembuatan databasenya :

1. Buka software Xampp, kemudian klik start pada lilihan Apache dab MySQL, tunggu hingga keduatulisannnyáberwarna hijau. Dan selanjutnya klik Shell untuk masuk ke CMD Xampp



Gambar 3. 2 Tampilan XAMPP

2. Setelah masuk pada terminal atau Shell, Ketikkan command “cd mysql/bin” yang digunakan untuk memindahkan direktori ke folder "bin" dalam direktori "mysql" di sistem pada terminal.
3. Kemudian masukan command “mysql -u root” yang digunakan untuk masuk ke dalam MySQL shell menggunakan akun "root" sebagai pengguna atau user.

```
C:\WINDOWS\system32\cmd. x + v
Microsoft Windows [Version 10.0.22621.1413]
(c) Microsoft Corporation. All rights reserved.

C:\Users\habil>cd c:\xampp\mysql\bin

c:\xampp\mysql\bin>mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 8
Server version: 10.4.27-MariaDB mariadb.org binary distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> |
```

Gambar 3. 3 Perintah awal pada XAMPP

4. Masukkan command “show database;” untuk menampilkan database yang ada pada PC

```
MariaDB [(none)]> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql      |
| performance_schema |
| phpmyadmin |
| test       |
+-----+
5 rows in set (0.039 sec)
```

Gambar 3. 4 Menampilkan database

Untuk mengelola dan memanipulasi basis data kita bisa menggunakan himpunan bagian dari SQL (*Structured Query Language*) DDL (*Data Definition Language*) dan DML (*Data Manipulation Language*).

3.4 Tahapan Pembuatan DDL

DDL adalah sekumpulan perintah SQL untuk mendefinisikan skema database. Ini hanya berurusan dengan deskripsi skema database dan digunakan untuk membuat dan memodifikasi struktur objek database. Contoh pernyataan DDL termasuk CREATE, ALTER, dan DROP. Berikut ini adalah penggunaan DDL pada database :

1. Untuk membuat database baru gunakan perintah “create database nama_data” usahakan nama basis datanya tidak menggunakan spasi

```
MariaDB [(none)]> create database kasir_rawat_inap;
Query OK, 1 row affected (0.001 sec)
```

Gambar 3. 5 Membuat database baru

2. Masukkan perintah “show database;” untuk mengecek Kembali database yang telah dibuat

```
MariaDB [(none)]> show databases;
+-----+
| Database |
+-----+
| hospital |
| hotel    |
| information_schema |
| kasir_rawat_inap |
| kuliah   |
| mysql    |
| olshopkita |
| performance_schema |
| phpmyadmin |
| test     |
+-----+
10 rows in set (0.024 sec)
```

Gambar 3. 6 Menampilkan database baru

3. Untuk Memulai membuat tabel di dalam database kasir_rawat_inap, maka kita harus mengaktifkan terlebih dulu databasenya dengan menggunakan perintah “use nama_database;”

```
MariaDB [(none)]> use kasir_rawat_inap;  
Database changed  
MariaDB [kasir_rawat_inap]>
```

Gambar 3. 7 Mengaktifkan database

4. Berikut merupakan perintah untuk membuat tabel pasien yang fungsinya untuk menyimpan data tentang pasien:

create table nama_table (

Kolom_1 type(panjang),

Kolom_2 type(panjang),

.....

Kolom_n type(panjang));

dan tambahkan perintah “primary key(masukan primarykey)” untuk menentukan primary key dari tabel yang kita buat

```
MariaDB [kasir_rawat_inap]> create table pasien(  
-> id_pasien int not null,  
-> nama_pasien varchar(25) not null,  
-> usia varchar(3) not null,  
-> alamat varchar(30) not null,  
-> primary key (id_pasien));  
Query OK, 0 rows affected (0.010 sec)
```

Gambar 3. 8 Membuat tabel baru

5. Buat juga tabel lainnya sesuai kebutuhan

```
MariaDB [kasir_rawat_inap]> create table kamar(  
-> no_kamar int not null,  
-> nama_kamar varchar(10) not null,  
-> jumlah_kamar int not null,  
-> primary key (no_kamar));  
Query OK, 0 rows affected (0.011 sec)
```

Gambar 3. 9 Menambahkan tabel baru yaitu tabel kamar

```
MariaDB [kasir_rawat_inap]> create table administrasi(  
-> id_adm int not null,  
-> nama_adm varchar(10) not null,  
-> tagihan int not null,  
-> primary key (id_adm));  
Query OK, 0 rows affected (0.010 sec)
```

Gambar 3. 10 Menambahkan tabel baru yaitu tabel administrasi

6. Untuk melihat tabel yang telah dibuat maka kita bisa masukan perintah “show tables;”. Namun yang muncul merupakan judul dari tabel yang telah dibuat

```
MariaDB [kasir_rawat_inap]> show tables;
+-----+
| Tables_in_kasir_rawat_inap |
+-----+
| administrasi                |
| kamar                      |
| pasien                      |
+-----+
3 rows in set (0.001 sec)
```

Gambar 3. 11 Menampilkan nama tabel

7. Jika kita ingin melihat atribut pada setiap tabel kita bisa ketikkan perintah “desc nama_table;” nantinya akan muncul tabel yang telah kita masukan

```
MariaDB [kasir_rawat_inap]> desc administrasi;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| id_adm | int(11) | NO | PRI | NULL | |
| nama_adm | varchar(10) | NO | | NULL | |
| tagihan | int(11) | NO | | NULL | |
| id_pasien | int(11) | NO | | NULL | |
| no_kamar | int(11) | NO | | NULL | |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.009 sec)
```

Gambar 3. 12 Menampilkan isi pada tabel

8. Kemudian kita coba merubah dan memodifikasi data pada tabel dengan perintah “alter table nama_table modify nama_data type_data(panjang);” kemudian cek apakah sudah berubah apa belum

```
MariaDB [kasir_rawat_inap]> alter table pasien modify alamat varchar(40) not null;
Query OK, 0 rows affected (0.010 sec)
Records: 0 Duplicates: 0 Warnings: 0

MariaDB [kasir_rawat_inap]> desc pasien;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| id_pasien | int(11) | NO | PRI | NULL | |
| nama_pasien | varchar(25) | NO | | NULL | |
| usia | varchar(3) | NO | | NULL | |
| alamat | varchar(40) | NO | | NULL | |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.010 sec)
```

Gambar 3. 13 Merubah dan memodifikasi kolom

9. Namun jika kita ingin menambah kolom pada table bisa menggunakan perintah berikut “alter table nama_table add nama_kolom_baru type(panjang) after/before nama_kolom;”


```

MariaDB [kasir_rawat_inap]> alter table pasien add pekerjaan varchar(15) after alamat;
Query OK, 0 rows affected (0.015 sec)
Records: 0 Duplicates: 0 Warnings: 0

MariaDB [kasir_rawat_inap]> desc pasien;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| id_pasien | int(11) | NO | PRI | NULL | |
| nama_pasien | varchar(25) | NO | | NULL | |
| usia | varchar(3) | NO | | NULL | |
| alamat | varchar(40) | NO | | NULL | |
| pekerjaan | varchar(15) | YES | | NULL | |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.011 sec)

```

Gambar 3. 14 Menambah kolom pada tabel

10. Dan jika ingin menghapus kolom bisa juga menggunakan perintah “Alter table nama_table drop nama_kolom;”

```

MariaDB [kasir_rawat_inap]> alter table pasien drop pekerjaan;
Query OK, 0 rows affected (0.009 sec)
Records: 0 Duplicates: 0 Warnings: 0

MariaDB [kasir_rawat_inap]> desc pasien;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| id_pasien | int(11) | NO | PRI | NULL | |
| nama_pasien | varchar(25) | NO | | NULL | |
| usia | varchar(3) | NO | | NULL | |
| alamat | varchar(40) | NO | | NULL | |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.013 sec)

```

Gambar 3. 15 Menghapus kolom

11. Untuk merubah nama dari tabelnya kita bisa menggunakan perintah “alter table nama_table rename nama_table_baru;”

```

MariaDB [kasir_rawat_inap]> alter table pasien rename tabel_pasien;
Query OK, 0 rows affected (0.008 sec)

MariaDB [kasir_rawat_inap]> show tables;
+-----+
| Tables_in_kasir_rawat_inap |
+-----+
| administrasi |
| kamar |
| tabel_pasien |
+-----+
3 rows in set (0.001 sec)

```

Gambar 3. 16 Rename tabel

12. Untuk menghapus tabel, kita bisa menggunakan perintah “drop table nama_table;”

```

MariaDB [kasir_rawat_inap]> show tables;
+-----+
| Tables_in_kasir_rawat_inap |
+-----+
| administrasi                |
| kamar                      |
| pegawai                    |
| tabel_pasien                |
+-----+
4 rows in set (0.001 sec)

MariaDB [kasir_rawat_inap]> drop table pegawai;
Query OK, 0 rows affected (0.007 sec)

MariaDB [kasir_rawat_inap]> show tables;
+-----+
| Tables_in_kasir_rawat_inap |
+-----+
| administrasi                |
| kamar                      |
| tabel_pasien                |
+-----+
3 rows in set (0.000 sec)

```

Gambar 3. 17 Menghapus tabel

13. Dan jika ingin menghapus databasenya bisa Menggunakan peintah “drop table nama_database”

```

+-----+
| Database |
+-----+
| hospital |
| hotel    |
| information_schema |
| kasir1   |
| kasir_rawat_inap |
| kuliah  |
| mysql    |
| olshopkita |
| performance_schema |
| phpmyadmin |
| test     |
+-----+
11 rows in set (0.001 sec)

MariaDB [(none)]> drop database kasir1;
Query OK, 0 rows affected (0.001 sec)

MariaDB [(none)]> show databases;
+-----+
| Database |
+-----+
| hospital |
| hotel    |
| information_schema |
| kasir_rawat_inap |
| kuliah  |
| mysql    |
| olshopkita |
| performance_schema |
| phpmyadmin |
| test     |
+-----+
10 rows in set (0.001 sec)

```

Gambar 3. 18 Menghapus databse

3.5 Data Manipulation Language

DML atau Data Manipulation Language adalah salah satu jenis bahasa pemrograman yang digunakan untuk mengelola atau memanipulasi data dalam sebuah database. Dalam DML, terdapat beberapa perintah seperti select, insert, update, dan delete yang digunakan untuk mengambil, menambah, mengubah, dan menghapus data dalam database. DML

digunakan oleh pengembang dan administrator database untuk memanipulasi data agar sesuai dengan kebutuhan aplikasi yang akan dijalankan. Penting untuk memahami dan menguasai DML agar dapat mengelola data dengan efektif dan efisien dalam sebuah database.

Perintah-perintah yang terdapat dalam DML :

1) Insert

Perintah insert digunakan untuk menyisipkan data baru ke dalam sebuah tabel database. Ada dua macam perintah insert, yaitu yang digunakan untuk menyisipkan data satu persatu dengan perintah 'insert into nama_table (variable)', dan yang menyisipkan banyak data sekaligus dengan perintah 'insert into nama_table (variable) values'. Di sini kami memilih yang sekaligus karena lebih mudah dan lebih cepat.

```
MariaDB [kasir_rawat_inap]> insert into pasien (id_pasien, nama_pasien, usia, alamat) values
-> ('0001','Nazril Irham','34','Jl. Bandung'),
-> ('0002','Liam Gallagher','56','Jl. Manchester'),
-> ('0003','John Winston Lennon','40','Jl. Liverpool');
Query OK, 3 rows affected (0.004 sec)
Records: 3 Duplicates: 0 Warnings: 0
```

Gambar 3. 19 Perintah insert

2) Select

Perintah select pada MySQL digunakan untuk menampilkan atau memunculkan isi data pada satu tabel atau lebih. Dalam perintah select dapat menggunakan perintah **select kolom1, kolom2, ..., kolom_n from nama_tabel;** dan dengan menggunakan perintah **select * from nama_tabel;** .

```
MariaDB [kasir_rawat_inap]> select * from pasien;
+-----+-----+-----+-----+
| id_pasien | nama_pasien | usia | alamat |
+-----+-----+-----+-----+
| 1 | Nazril Irham | 34 | Jl. Bandung |
| 2 | Liam Gallagher | 56 | Jl. Manchester |
| 3 | John Winston Lennon | 40 | Jl. Liverpool |
+-----+-----+-----+-----+
3 rows in set (0.000 sec)
```

Gambar 3. 20 Perintah select

3) Update

Perintah UPDATE digunakan untuk memodifikasi atau memperbaharui nilai-nilai record pada sebuah tabel. Sintaksnya adalah sebagai berikut: **UPDATE nama_tabel SET field1 = nilai1, field2 = nilai2 , ..., field_n = nilai_n WHERE kondisi;**

Untuk melakukan perubahan struktur tabel pada tabel pasien, maka anda dapat menggunakan perintah: **update pasien set id_pasien = '2100001' where id_pasien='0001';**

```

MariaDB [kasir_rawat_inap]> update pasien set id_pasien ='2100001' where id_pasien='0001';
Query OK, 1 row affected (0.003 sec)
Rows matched: 1 Changed: 1 Warnings: 0

MariaDB [kasir_rawat_inap]> select * from pasien;
+-----+-----+-----+-----+
| id_pasien | nama_pasien | usia | alamat |
+-----+-----+-----+-----+
| 2 | Liam Gallagher | 56 | Jl. Manchester |
| 3 | John Winston Lennon | 40 | Jl. Liverpool |
| 2100001 | Nazril Irham | 34 | Jl. Bandung |
+-----+-----+-----+-----+
3 rows in set (0.000 sec)

```

Gambar 3. 21 Perintah update

4) Delete

Perintah DELETE digunakan untuk menghapus record pada suatu tabel. Sintaks dari perintah DELETE adalah sebagai berikut : **delete from nama_tabel where kriteria;**

Hati-hati juga dalam memberikan perintah delete ini, karena jika lupa memberikan pernyataan kondisi, perintah ini akan menyebabkan terhapusnya seluruh isi tabel.

Setelah itu, bisa dicek isi tabel nya, apakah sudah terhapus atau belum menggunakan perintah 'select * from pasien;'

```

MariaDB [kasir_rawat_inap]> delete from pasien where id_pasien='2100001';
Query OK, 1 row affected (0.003 sec)

MariaDB [kasir_rawat_inap]> select * from pasien;
+-----+-----+-----+-----+
| id_pasien | nama_pasien | usia | alamat |
+-----+-----+-----+-----+
| 2100002 | John Winston Lennon | 40 | Jl. Liverpool |
| 2100003 | Liam Gallagher | 56 | Jl. Manchester |
+-----+-----+-----+-----+
2 rows in set (0.000 sec)

```

Gambar 3. 22 Perintah delete

3.6 Implementasi CRUD

Code yang kami gunakan menggunakan Bahasa pemograman Javascript dengan library Node.js dan berikut implementasi Create Read Update Delete yang telah kami gunakan.

1) Create

Pada bagian ini add digunakan untuk menambah list pasien, dari mulai ID, Pasien Name,Usia,Alamat

ID	Pasien Name	Usia	Alamat	
1231	Wadaw	21	Ubudu	Edit Delete
2100001	Nazil Iham	34	Jl. Bandung	Edit Delete
2100002	John Winston Lennon	40	Jl. Liverpool	Edit Delete
2100003	Liam Gallagher	36	Jl. Manchester	Edit Delete

Gambar 3. 23 Implementasi Create

Pasien Add

Pasien ID

Pasien Name

Usia

Alamat

[Save Pasien](#)

Gambar 3. 24 Implementasi Create

2) Read

Pada bagian Read disini kami menggunakan untuk menampilkan sesuatu list, yang sudah di Add Delete dan Update.

Ada 3 tampilan yang kami gunakan dari mulai Details pasient, Administrasi. Dan Room(list kamar)

Pasiem Details

ID	Pasien Name	Usia	Alamat	
1231	Wadaw	21	Ubudu	Edit Delete
2100001	Nazil Iham	34	Jl. Bandung	Edit Delete
2100002	John Winston Lennon	40	Jl. Liverpool	Edit Delete
2100003	Liam Gallagher	36	Jl. Manchester	Edit Delete

Administrasi

ID adm	tagihan	nama pasien	Alamat	no kamar	nama kamar
11	1200000	Nazil Iham	Jl. Bandung	212	Ruangan1
12	3200000	John Winston Lennon	Jl. Liverpool	213	Ruangan2
13	2000000	Liam Gallagher	Jl. Manchester	214	Ruangan3

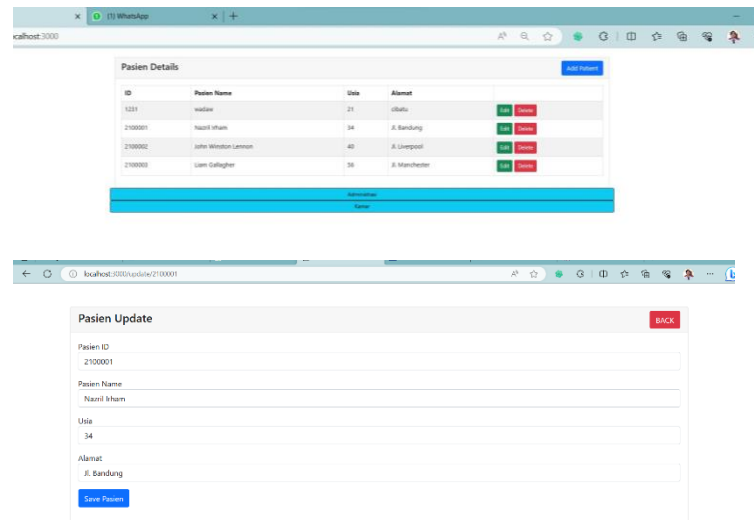
Room

ID adm	tagihan	nama pasien
212	Ruangan1	3
213	Ruangan2	3
214	Ruangan3	2

Gambar 3. 25 Implementasi Read

3) Update

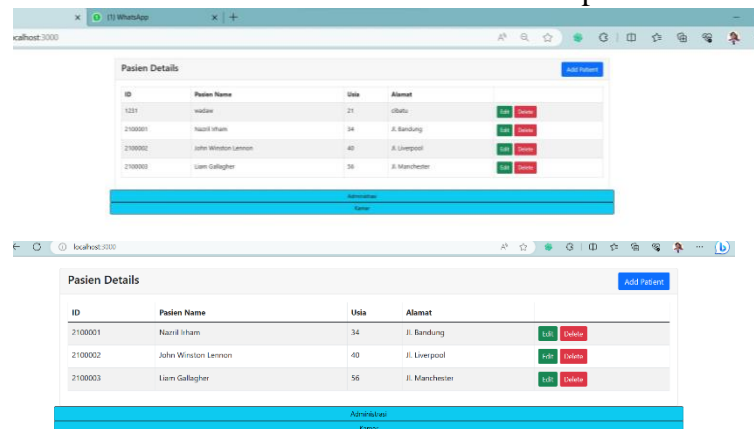
Pada bagian update ini digunakan untuk mengubah data yang sudah tertera sebelumnya, data tersebut akan otomatis terganti dengan data yang baru.



Gambar 3. 26 Implementasi Update

4) Delete

Delete disini digunakan untuk menghapus salah satu list dari tabel yang sudah tertera, Ketika tombol delete ditekan otomatis list dari tabel tersebut terhapus



Gambar 3. 27 Implementasi Delete

3.7 Implementasi Seluruh Syntax

1) App

```
const express = require('express');
const mysql = require('mysql');
const app = express();
app.use(express.static('public'));
```

```

app.use(express.urlencoded({extended:false}));
const connection = mysql.createConnection({
  host: 'localhost',
  user: 'root',
  password: '',
  database: 'kasir_rawat_inap',
  port:3306
});
connection.connect((err) => {
  if (err) {
    console.error('Kesalahan koneksi:', err);
    return;
  }
  console.log('Terhubung ke database MySQL');
});
app.get('/',(req,res)=> {
  connection.query('SELECT * FROM
pasien',(error,result)=> {
    res.render('index.ejs',{data:result});
  })
});
app.get('/room',(req,res)=> {
  connection.query('SELECT * FROM
kamar',(error,result)=> {
    res.render('room.ejs',{data:result});
  })
});
app.get('/delete/:id',(req,res)=> {
  connection.query('SELECT * FROM
pasien',(error,result)=> {
    connection.query('DELETE FROM pasien WHERE
id_pasien=?',[req.params.id],[error,resultDelete]=> {
      res.redirect('/')
    })
  })
});
app.get('/add',(req,res)=> {
  res.render('add.ejs');
});
app.post('/create',(req,res)=> {
  connection.query('insert into pasien
(id_pasien,nama_pasien,usia,alamat) value
(?,?,?,?)',[req.body.id,req.body.nama,req.body.usia,req.body.alamat],[errors,result]=> {
    res.redirect('/');
  })
});

```

```

app.get('/update/:id',(req,res)=> {
    connection.query('SELECT * FROM pasien WHERE
id_pasien=?',[req.params.id] ,(error,result)=> {
        // console.log(result);
        res.render('update.ejs',{data:result[0]});
    })
});
app.post('/change/:id',(req,res)=> {
    connection.query('UPDATE pasien SET
id_pasien=?,nama_pasien=?,usia=?,alamat=? WHERE
id_pasien=?',[req.body.id,req.body.nama,req.body.usia,
req.body.alamat,req.params.id] ,(error,result)=> {
        res.redirect('/');
    })
});
app.get('/kamar',(req,res)=>{
    connection.query('SELECT administrasi.id_adm,
administrasi.tagihan, pasien.nama_pasien,
pasien.alamat, kamar.no_kamar,kamar.nama_kamar FROM
administrasi, pasien,kamar WHERE
administrasi.id_pasien= pasien.id_pasien &&
administrasi.no_kamar=kamar.no_kamar', (error,result)=>
{
    res.render('kamar.ejs',{data:result});
})
// app.get('/room',(req,res)=> {
//     connection.query('SELECT *from
kamar', (error,result)=>{
//         res.render('room.ejs',{data:result});
//     })

//     app.get('/room',(req,res)=> {
//         connection.query('SELECT * FROM
kamar', (error,result)=> {
//             res.render('room.ejs',{data:result});
//         })
//     })
});

```



```
app.listen(3000,()=> {  
  console.log('port berjalan')});
```

2) Index

```
<!DOCTYPE html>  
<html lang="en">  
<head>  
  <meta charset="UTF-8">  
  <meta http-equiv="X-UA-Compatible"  
content="IE=edge">  
  <meta name="viewport" content="width=device-width,  
initial-scale=1.0">  
  <meta charset="utf-8">  
  <meta name="viewport" content="width=device-width,  
initial-scale=1">  
  
  <!-- Bootstrap CSS -->  
  <link  
href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dis  
t/css/bootstrap.min.css" rel="stylesheet">  
  
  <title>Pasien CRUD</title>  
</head>  
  
<body>  
  
  <div class="container mt-4">  
    <div class="row">  
      <div class="col-md-12">  
        <div class="card">  
          <div class="card-header">  
            <h4>Pasien Details  
            <a href="/add"  
class="btn btn-primary float-end">Add Patient</a>  
          </h4>  
          </div>  
          <div class="card-body">  
            <table class="table table-  
bordered table-striped">  
  
              <thead>
```

```

                <tr>
                    <th>ID</th>
                    <th>Pasien
Name</th>
                    <th>Usia</th>
                    <th>Alamat</th>
                <th></th>
            </tr>
        </thead>
        <tbody>
            <%
data.forEach((pasien)=> { %>
                <tr>
                    <td><%=pasien.
id_pasien%></td>
                    <td><%=pasien.
nama_pasien%></td>
                    <td><%=pasien.
usia%></td>
                    <td><%=pasien.
alamat%></td>
                    <td>
                        <a
href="update/<%= pasien.id_pasien %>" class="btn btn-
success btn-sm">Edit</a>
                        <a
href="delete/<%= pasien.id_pasien %>" class="btn btn-
danger btn-sm">Delete</a>
                    </td>
                </tr>
            <%}) %>
        </tbody>
    </table>
</div>
</div>
</div>
    <a href="/kamar" class="btn btn-info
btn-sm border border-dark">Administrasi</a>
    <a href="/room" class="btn btn-info
btn-sm border border-dark">Kamar</a>
</div>
</div>

```

```

        <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist
/js/bootstrap.bundle.min.js"></script>

    </body>
</html>

```

3) Kamar

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible"
content="IE=edge">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1">

    <!-- Bootstrap CSS -->
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dis
t/css/bootstrap.min.css" rel="stylesheet">

    <title>Pasien CRUD</title>
</head>
<body>
    <div class="container mt-4">
        <div class="row">
            <div class="col-md-12">
                <div class="card">
                    <div class="card-header">
                        <h4>Administrasi

                    </h4>
                    </div>
                    <div class="card-body">
                        <table class="table table-
bordered table-striped">
                            <thead>

```

```

                <tr>
                    <th>ID adm</th>
                    <th>tagihan</th>
                    <th>nama
pasien</th>
                    <th>Alamat</th>
                    <th>no
kamar</th>
                    <th>nama
kamar</th>

                </tr>
            </thead>
            <tbody>
                <%
data.forEach((pasien)=> { %>
                    <tr>
                        <td><%=pasien.id_a
dm%></td>
                        <td><%=pasien.tagi
han%></td>
                        <td><%=pasien.nama
_pasien%></td>
                        <td><%=pasien.alam
at%></td>
                        <td><%=pasien.no_k
amar%></td>
                        <td><%=pasien.nama
_kamar%></td>

                    </tr>
                <%}) %>
            </tbody>
        <tbody>
            <tr>
                <td></td>
            </tr>
        </tbody>
    </table>
</div>
</div>
</div>
</div>
</div>

```

```
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist
/js/bootstrap.bundle.min.js"></script>
</body>
</html>
```

4) Room

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible"
content="IE=edge">
  <meta name="viewport" content="width=device-width,
initial-scale=1.0">
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width,
initial-scale=1">

  <!-- Bootstrap CSS -->
  <link
href="https://cdn.jsdelivrivr.net/npm/bootstrap@5.1.3/dis
t/css/bootstrap.min.css" rel="stylesheet">

  <title>Pasien CRUD</title>
</head>
<body>
  <div class="container mt-4">
    <div class="row">
      <div class="col-md-12">
        <div class="card">
          <div class="card-header">
            <h4>Room

            </h4>
          </div>
          <div class="card-body">
            <table class="table table-
bordered table-striped">

              <thead>
                <tr>
                  <th>ID adm</th>
                  <th>tagihan</th>
                  <th>nama
pasien</th>
```

```

        </tr>
    </thead>
    <tbody>
        <%
data.forEach((pasien)=> { %>
            <tr>
                <td><%=pasien.no_k
amar%></td>
                <td><%=pasien.nama
_kamar%></td>
                <td><%=pasien.juml
ah_kamar%></td>

            </tr>
        <%}) %>
    </tbody>
</tbody>
    <tr>
        <td></td>
    </tr>
</tbody>
</table>
</div>
</div>
</div>
</div>
</div>
</div>

<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist
/js/bootstrap.bundle.min.js"></script>
</body>
</html>

```

5) Add

```

<?php
session_start();
?>

<!doctype html>
<html lang="en">

```

```

<head>
  <!-- Required meta tags -->
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width,
initial-scale=1">

  <!-- Bootstrap CSS -->
  <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dis
t/css/bootstrap.min.css" rel="stylesheet">

  <title>Pasien Create</title>
</head>
<body>

  <div class="container mt-5">
    <div class="row">
      <div class="col-md-12">
        <div class="card">
          <div class="card-header">
            <h4>Pasien Add
              <a href="/" class="btn
btn-danger float-end">BACK</a>
            </h4>
          </div>
          <div class="card-body">
            <form action="/create"
method="POST">
              <div class="mb-3">
                <label>Pasien
ID</label>
                <input type="text"
name="id" class="form-control">
              </div>
              <div class="mb-3">
                <label>Pasien
Name</label>
                <input type="text"
name="nama" class="form-control">
              </div>
              <div class="mb-3">
                <label>Usia</label>
                <input type="text"
name="usia" class="form-control">
              </div>
              <div class="mb-3">
                <label>Alamat</label>

```

```

        <input type="text"
name="alamat" class="form-control">
    </div>
    <div class="mb-3">
        <button type="submit"
name="save_pasien" class="btn btn-primary">Save
Pasien</button>
    </div>
</form>
</div>
</div>
</div>
</div>
</div>
</div>
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist
/js/bootstrap.bundle.min.js"></script>
</body>
</html>

```


BAB IV

KESIMPULAN

Dari praktikum kali ini kesimpulannya adalah selain dari memahami lebih lanjut mengenai basis data, untuk membangun basis data yang baik, diperlukan tahapan rancangan yang terstruktur dan metodologi yang tepat. Salah satu tahapan penting dalam rancangan basis data adalah pembuatan ERD yang dapat membantu merancang struktur data dengan lebih baik. Selain itu dalam memilih model basis data, perlu dipertimbangkan kebutuhan aplikasi dan karakteristik data yang akan disimpan.

DAFTAR PUSTAKA

Silberchartz, A., Korth, H., & Sudhharsan, S. (2011). Database System Concept.
Database System Concept SIXth Edition

https://elearning.itg.ac.id/student_area/materi/detail/7834

Utomo , W. (2010). Pemodelan Basis Data Berbasis Objek.








LAMPIRAN

ASISTENSI TUGAS PRAKTIKUM BASIS DATA

Nama Pembimbing : Riri Asri Indah Pertiwi

Kelompok : 9

Anggota Kelompok & Tugas : Habil Gymnastiar Abdul Matin (2106008) Kasir Rawai Ima
Sahrudin Fiqri Muzahidat (2106024)

No.	Tanggal	Uraian Asistensi	Tanda Tangan
1.	09-05-2023	Konsultasi Rancangan ERD	
2.	16-05-2023	Perbaikan rancangan ERD	
3.	23-05-2023	Final rancangan ERD	
4.	30-05-2023	Pembuatan DDL	
5.	6-06-2023	Pembuatan DML	
6.	13-06-2023	Rancangan tampilan	
7.	14-06-2023	Implementasi sintaks MySQL ke dalam bahasa pemrograman (CRUD)	
8.			
9.			
10.			

*Catatan :

1. Lembar Asistensi harus diserahkan kepada pembimbing pada saat proses asistensi.
2. Minimal melakukan asistensi dengan pembimbing sebanyak 7 kali pertemuan.
3. Cantumkan bagian keterangan pembagian tugas di bagian kolom "Uraian Asistensi". Contoh Pembuatan Program (Reza)".

Instruktur Praktikum



Detila Rostilawati, S. Kom