

LAPORAN
PRAKTIKUM PEMOGRAMAN BERORIENTASI OBJECT



OLEH :

NAMA : RIZKI
NIM : F1G120009
KELOMPOK : 1 (SATU)

ASISTEN PENGAMPU:
WAHID SAFRI JAYANTO

PROGRAM STUDI S1 ILMU KOMPUTER
JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS HALU OLEO
KENDARI
2021

HALAMAN PENGESAHAN
LAPORAN PRAKTIKUM



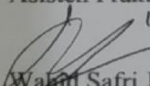
OLEH:


RIZKI (F1G120009)

Laporan praktikum Pemrograman Berorientasi Object ini disusun sebagai tugas akhir menyelesaikan praktikum Pemrograman Berorientasi Object sebagai salah satu syarat lulus matakuliah Pemrograman Berorientasi Object. Menerangkan bahwa yang tertulis dalam laporan lengkap ini adalah benar dan dinyatakan telah memenuhi syarat.

Kendari , 12 Desember 2021

Menyetujui

Asisten Praktikum
12-12-2021

Wafid Safri Jayanto
F1G117059

Praktikan

Rizki
F1G120009

KATA PENGANTAR

Dengan menyebut nama Allah SWT yang Maha Pengasih lagi Maha Penyayang. Kami panjatkan puja dan puji syukur kehadirat-Nya. Yang telah melimpahkan rahmat dan hidayahnya kepada kami, sehingga kami dapat menyelesaikan Laporan Praktikum PBO ini.

Adapun laporan ini kami telah usahakan semaksimal mungkin dan tentunya dengan bantuan berbagai pihak, sehingga dapat memperlancar pembuatan laporan ini. Untuk itu kami tak lupa pula menyampaikan banyak terimakasih kepada semua pihak yang telah membantu dalam pembuatan laporan ini.

Namun tidak lepas dari semua itu kami menyadari sepenuhnya bahwa ada kekurangan baik dari segi penyusun bahasa dan segi lainnya. Oleh karena itu dengan lapang dada dan tangan terbuka kami membuka selebar-lebarnya bagi pembaca yang ingin memberi saran dan kritik kepada kami sehingga kami dapat memperbaiki laporan ini.

Akhirnya penyusun mengharapkan semoga dari laporan praktikum PBO ini dapat diambil hikmah dan manfaatnya sehingga dapat memberikan inspirasi terhadap pembaca.

Kendari

Penyusun

DAFTAR ISI

COVER.....	i
HALAMAN PENGESAHAN	Error! Bookmark not defined.
KATA PENGANTAR	iii
DAFTAR ISI	iv
DAFTAR TABLE.....	vi
DAFTAR GAMBAR	vii
PRAKTIKUM 1.....	1
1.1.1 Waktuk dan tempat	1
1.1.2 Alat dan Bahan.....	1
1.1.3 Pengenalan PBO(<i>Pemograman Berorientasi Object</i>)	2
1.1.4 Pengenalan <i>PHP</i>	4
PRAKTIKUM 2.....	7
2.1.1 <i>Class</i>	7
2.1.2 <i>Method</i>	7
2.1.3 <i>Constructor</i>	8
2.1.4 <i>Property</i>	9
2.1.5 <i>Object</i>	10
2.1.6 <i>Modifier</i>	11
2.1.7 <i>Atribut</i>	11
2.1.8 <i>Composer</i>	12
2.1.9 <i>Laravel</i>	12

2.1.10 <i>Constructor</i> dan <i>Destructor</i>	13
2.1.11 <i>Interface</i>	14
PRAKTIKUM 3.....	16
3.1.1 Model data berbasis <i>object</i>	16
3.1.2 Model data berbasis <i>record</i>	17
3.1.3 Penjelasan crud	19
PRAKTIKUM 4.....	23
4.1.1 <i>Project</i> Membuat Sistem Penyewaan Kamar Kos	23
4.1.2 DFD (<i>Data Flow Diagram</i>)	24
4.1.3 <i>Interface</i>	26
DAFTAR PUSTAKA	33

DAFTAR TABLE

Tabel 1. 1 Alat dan bahan.....	1
--------------------------------	---

DAFTAR GAMBAR

Gambar 3.1 1 <i>Semantic model</i>	16
Gambar 3.1 2 ERD	17
Gambar 3.2 1 Model hirarki.....	17
Gambar 3.2 2 Model jaringan.....	18
Gambar 3.2 3 Model <i>relational</i>	18
Gambar 4.1 1 <i>Entity Relationship Diagram</i> Sistem Penyewaan Kamar kos.....	23
Gambar 4.2 1 Diagrama Flow level 0.....	26
Gambar 4.2 2 Diagram flow level 1	26
Gambar 4.3 1 <i>Login</i>	27
Gambar 4.3 2 Admin	28
Gambar 4.3 3 Daftar kamar kos.....	29
Gambar 4.3 4 Tambah Data kos	29
Gambar 4.3 5 Penyewaan Kamar Kos.....	30
Gambar 4.3 6 Penyewaan Kamar kos.....	31
Gambar 4.3 7 Halaman Penyewa	31

PRAKTIKUM 1

1.1.1 Waktu dan tempat

Kegiatan praktikum Pemograman Berorientasi Objek ini dimulai dari tanggal 30 September sampai 2 Desember dilaksanakan setiap hari kamis pukul 8.00-10.00 WITA di Laboratorium Aljabar lantai 3 gedung A, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Halu Oleo, Kendari

1.1.2 Alat dan Bahan

Adapun alat dan bahan yang digunakan dalam praktikum kali ini ialah:

No.	Alat dan Bahan	Kegunaan
	Laptop	perangkat keras (<i>hardware</i>) yang digunakan untuk menyimpan aplikasi
1.	Xampp	XAMPP digunakan untuk membuat web server lokal pada komputer
2.	Sublime Tex3	Digunakan sebagai tempat untuk membuat codingan
3.	Phpmyadmin	Digunakan untuk membuat database
4.	<i>Chrome</i>	Digunakan untuk menampilkan sebuah <i>project</i> yang kita buat

Tabel 1. 1 Alat dan bahan

1.1.3 Pengenalan PBO(*Pemograman Berorientasi Object*)

Pada pertemuan satu yang dibahas mengenai materi apa itu Pemrograman Berorientasi Objek. Pemrograman Berorientasi Objek (*Object Oriented Programming* atau OOP) merupakan paradigma pemrograman yang berorientasikan kepada objek. Objek adalah struktur data yang terdiri dari bidang data dan metode bersama dengan interaksi mereka untuk merancang aplikasi dan program komputer. Semua data dan fungsi di dalam paradigma ini dibungkus dalam kelas-kelas atau objek-objek. Bandingkan dengan logika pemrograman terstruktur. Setiap objek dapat menerima pesan, memproses data, dan mengirim pesan ke objek lainnya. Pada jaman sekarang, banyak bahasa pemrograman yang mendukung OOP.

OOP adalah paradigma pemrograman yang cukup dominan saat ini, karena mampu memberikan solusi kaidah pemrograman modern. Meskipun demikian, bukan berarti bahwa pemrograman prosedural sudah tidak layak lagi. OOP diciptakan karena dirasakan masih adanya keterbatasan pada bahasa pemrograman tradisional. Konsep dari OOP sendiri adalah semua pemecahan masalah dibagi ke dalam objek. Dalam OOP data dan fungsi-fungsi yang akan mengoperasikannya digabungkan menjadi satu kesatuan yang dapat disebut sebagai objek. Proses perancangan atau desain dalam suatu pemrograman merupakan proses yang tidak terpisah dari proses yang mendahului, yaitu analisis dan proses yang mengikutinya. Pembahasan mengenai orientasi objek tidak akan terlepas dari konsep objek seperti *inheritance* atau penurunan, *encapsulation* atau pembungkusan, dan *polymorphism* atau

kebanyak rupa. Konsep-konsep ini merupakan fundamental dalam orientasi objek yang perlu sekali dipahami serta digunakan dengan baik, dan menghindari penggunaannya yang tidak tepat.

Model data berorientasi objek dikatakan dapat memberi fleksibilitas yang lebih, kemudahan mengubah program, dan digunakan luas dalam teknik piranti lunak skala besar. Lebih jauh lagi, pendukung OOP mengklaim bahwa OOP lebih mudah dipelajari bagi pemula dibanding dengan pendekatan sebelumnya, dan pendekatan OOP lebih mudah dikembangkan dan dirawat.

Package adalah suatu cara pengelompokan dan pengorganisasian kelas-kelas kedalam suatu *library*. *Package* bekerja dengan membuat direktori dan *folder* baru sesuai dengan penamaan *package*, kemudian menyimpan file *class* pada *folder* tersebut. Deklarasi *package* pada baris paling atas sebelum perintah *import*. Kelas merupakan bagian utama pada pemrograman java, kelas merupakan hierarki tertinggi dari bahasa java, dimana di dalam *body* kelas ini didefinisikan *variable*, *method*, dan kelas *inner*. Deklarasi kelas otomatis terbentuk saat membuat file java baru, kemudian ditambahkan secara manual *modifier*, pewarisan (*extends*), dan *interface* (*implements*).

Perintah *import* digunakan untuk memberitahukan kepada program untuk mengacu pada kelas-kelas yang terdapat pada *package* tersebut bukan menjalankan kelas-kelas tersebut. Dalam program, dapat *mengimport* hanya kelas tertentu dapat pula *mengimport* semua kelas menggunakan tanda asterisk (*) pada akhir nama

package. Sedangkan untuk *mengimport* kelas tertentu, dapat menuliskan nama kelas setelah nama *package*.

Method adalah bagian program yang menjelaskan tingkah laku dari *object* yang akan di-*instance*. *Method* tidak dapat berdiri sendiri sebagaimana kelas, di mana letak penulisan berada didalam *body* kelas. *Method* berdasarkan jenisnya dabagi menjadi beberapa kategori yaitu:

Konstruktor

Konstruktor adalah *method* yang dieksekusi pertama sekali setelah *method* main. Biasanya *method* konstruktor digunakan untuk memberikan nilai inisialisasi program. Nama dari *method* konstruktor harus sama dengan nama kelas.

Fungi/Prosedur

Fungsi adalah *method* yang mengembalikan sebuah nilai, sedangkan prosedur adalah *method* yang tidak mengembalikan sebuah nilai Main *Method* main adalah *method* utama yang pertama kali dipanggil untuk menjalankan program. Sebuah program yang tidak mempunyai *method* main tidak akan bisa dijalanka atau dieksekusi.

1.1.4 Pengenalan *PHP*

Pada awalnya *PHP* merupakan kependekan dari *Personal Home Page* (Situs personal). *PHP* pertama kali dibuat oleh Rasmus Lerdorf pada tahun 1995. Pada waktu itu *PHP* masih bernama *Form Interpreted* (FI), yang wujudnya berupa sekumpulan skrip yang digunakan untuk mengolah data formulir dari web.

Selanjutnya Rasmus merilis kode sumber tersebut untuk umum dan menamakannya PHP/FI. Dengan perilsan kode sumber ini menjadi sumber terbuka, maka banyak pemrogram yang tertarik untuk ikut mengembangkan PHP.

Pada November 1997, dirilis PHP/FI 2.0. Pada rilis ini, *interpreter* PHP sudah diimplementasikan dalam program C. Dalam rilis ini disertakan juga modul-modul ekstensi yang meningkatkan kemampuan PHP/FI secara signifikan.

Pada tahun 1997, sebuah perusahaan bernama Zend menulis ulang *interpreter* PHP menjadi lebih bersih, lebih baik, dan lebih cepat. Kemudian pada Juni 1998, perusahaan tersebut merilis *interpreter* baru untuk PHP dan meresmikan rilis tersebut sebagai PHP 3.0 dan singkatan PHP diubah menjadi akronim berulang PHP: *Hypertext Preprocessing*.

Pada pertengahan tahun 1999, Zend merilis *interpreter* PHP baru dan rilis tersebut dikenal dengan PHP 4.0. PHP 4.0 adalah versi PHP yang paling banyak dipakai pada awal abad ke-21. Versi ini banyak dipakai disebabkan kemampuannya untuk membangun aplikasi web kompleks tetapi tetap memiliki kecepatan dan stabilitas yang tinggi.

Pada Juni 2004, Zend merilis PHP 5.0. Dalam versi ini, inti dari *interpreter* PHP mengalami perubahan besar. Versi ini juga memasukkan model pemrograman berorientasi objek ke dalam PHP untuk menjawab perkembangan bahasa pemrograman ke arah paradigma berorientasi objek. Server web bawaan ditambahkan pada versi 5.4 untuk mempermudah pengembang menjalankan kode PHP tanpa

menginstall *software server*. Versi terbaru dan stabil dari bahasa pemrograman PHP saat ini adalah versi 7.0.16 dan 7.1.2 yang resmi dirilis pada tanggal 17 Februari 2017.

a. Pengertian PHP

PHP adalah singkatan dari “PHP: *Hypertext Preprocessor*”, yaitu bahasa pemrograman disisi *server* yang digunakan secara luas untuk penanganan pembuatan dan pengembangan sebuah situs web dan bisa digunakan bersamaan dengan HTML. Ketika Anda mengakses sebuah URL, maka web browser akan melakukan *request* ke sebuah web server.

b. Cara penulisan *syntax php*

untuk penulisan *syntaxnya*, php di tandai dengan membuat tag pembuka (`<?php`) dan di akhiri dengan tag penutup (`<?`). *syntax php* dapat disisipkan pada bagian-bagian html. kemudia di akhir setiap baris *syntax php* harus di tutup dengan tanda semicolom atau titik koma (;) berikut ini adalah contoh penulisan *syntax php* yang benar

```
<?php
echo "Belajar Pemrograman PHP ";
?>
```

PRAKTIKUM 2

2.1.1 Class

class didalam oop digunakan untuk membuat sebuah kerangka kerja. bisa dikatakan sebagai *library*. *class* berisi *property* dan *method*. jadi ibaratnya *class* adalah sebuah wadah yang menyimpan *property* dan *method* dan juga *object* yang dihasilkan biasanya berdasarkan isi dari *class*

```
<?php

//Cara penulisan class OOP PHP
class nama_class{

    //isi dari class ini
}

?>
```

2.1.2 Method

method adalah sebuah aksi yang terdapat didalam *class*. penulisan *method* pada *class* oop adalah dengan menuliskan *syntax function* diawalnya lalu di ikuti dengan nama method tersebut.

Berikut contoh penulisan method pada oop php

```
<?php

//Cara penulisan class dan property OOP PHP
class mobil{
    // property oop
    var $warna;
    var $merek;
    var $ukuran;

    //method oop
    function maju(){
        //isi method
    }
    function berhenti(){
        //isi mehod
    }
}

?>
```

2.1.3 Constructor

Constructor adalah *method* khusus yang akan dijalankan secara otomatis pada saat sebuah objek dibuat (*instansiasi*), yakni ketika perintah “new” dijalankan. *Constructor* biasa digunakan untuk membuat proses awal dalam mempersiapkan objek, seperti memberi nilai awal kepada *property*, memanggil *method* internal dan beberapa proses lain yang digunakan untuk mempersiapkan objek. Dalam PHP, constructor dibuat menggunakan *method* `__construct()`.

Contoh *syntax* :

```
<?php

class Mahasiswa {
    public function __construct() { // Contoh Constructor
        echo "Fungsi Construct Terpanggil";
    }
    public function __destruct() { // Contoh Desstructor
        echo "Fungsi Destruct Terpanggil";
    }
}

// function __construct() terpanggil ketika objek ini dibuat
$rizal = new Mahasiswa(); // Object
echo "<br/>Batas <br/>";
// function __destruct() terpanggil ketika file berakhir

?>
```

2.1.4 Property

property adalah data-data yang terdapat didalam *class*. datanya bisa berupa sifat. kegunaan *property* pada sebuah *class* sama dengan kegunaan variabel di php bisa digunakan untuk menyimpan data dan lain lain. cara penulisan *property* pada *class* adalah dengan diawali *syntax var*. cara penamaan *property* sama dengan aturan penamaan variable

syntax property

```
<?php

//Cara penulisan class dan property OOP PHP

class mobil{

    var $warna;
    var $merek;
    var $ukuran;

}

?>
```

2.1.5 Object

Object adalah *output* dan *class* dan *object* juga dapat menampilkan atau mengelola isi *class*. seluruh isi *class* akan kita instansiasikan menjadi *object*

contoh penulisan *object*

```
<?php

//Cara penulisan class dan property OOP PHP
class mobil{

    //isi class

}

$mobil = new mobil();

?>
```

2.1.6 Modifier

Modifier adalah kata, *phrase* , atau *clause* yang berfungsi sebagai *adjective* atau *adverb* yang menerangkan kata atau kelompok kata lain. Sebagai *adjective* dan *adverb* ketika berfungsi sebagai *adjective* (dapat berupa *simple adjective*, *adjective phrase*, *clause participle*, *infinitive*), *modifier* menerangkan *noun*, sedangkan ketika berfungsi sebagai *adverb* (dapat berupa *simple adverb* , *adverb phrase*, *clause*, *preposition phrase*, *infinitive*), kata ini menerangkan *verb*, *adjective* atau *adverb* lain. .
(Gunadarman, 2013)

Contoh Program:

```
Public class bank balance
{
    public String owner
    public int balance
    public bank_balance(String name, int dollars )
    {
        owner = name;
        if(dollars > = 0)
```

2.1.7 Atribut

Atribut merupakan nilai data yang terdapat pada suatu *object* di dalam *class*. *Attribute* mempunyai karakteristik yang membedakan *object* yang satu dengan *object* yang lainnya. Contoh : pada *Class* Buah terdapat *attribute*:warna, berat.

Misalkan pada *object* mangga: warna berisi kuning dan berat 0.5 kg dan pada *object* apel : warna merah dan berat 0.6 kg (Andre 2015).

2.1.8 Composer

Composer adalah *package-manager* (di level aplikasi) untuk bahasa pemrograman PHP. Menawarkan standarisasi cara pengelolaan *libraries* dan *software dependencies* dalam projek PHP. *Composer* memungkinkan kita mendefinisikan pustaka atau *library* apa saja yang projek kita butuhkan, untuk kemudian *Composer* lah yang akan menangani proses instalasi dan penyiapan pustaka-pustaka tersebut untuk kita gunakan

Cara Penggunannya:

```
<?php
// misalkan ini adalah file index.php

require_once __DIR__ . '/vendor/autoload.php';

$fb = new \Facebook\Facebook([
    'app_id' => '{app-id}',
    'app_secret' => '{app-secret}',
    'default_graph_version' => 'v2.10',
    //'default_access_token' => '{access-token}', // optional
]);
```

2.1.9 Laravel

Laravel diluncurkan sejak tahun 2011 dan mengalami pertumbuhan yang cukup eksponensial. Di tahun 2015, Laravel adalah *framework* yang paling banyak mendapatkan bintang di Github. Sekarang *framework* ini menjadi salah satu yang populer di dunia, tidak terkecuali di Indonesia.

Laravel fokus di bagian end-user, yang berarti fokus pada kejelasan dan kesederhanaan, baik penulisan maupun tampilan, serta menghasilkan fungsionalitas aplikasi web yang bekerja sebagaimana mestinya. Hal ini membuat *developer* maupun perusahaan menggunakan *framework* ini untuk membangun apa pun, mulai dari proyek kecil hingga skala perusahaan kelas atas.

Laravel mengubah pengembangan website menjadi lebih elegan, ekspresif, dan menyenangkan, sesuai dengan jargonnya “*The PHP Framework For Web Artisans*”. Selain itu, Laravel juga mempermudah proses pengembangan *website* dengan bantuan beberapa fitur unggulan, seperti *Template Engine*, *Routing*, dan *Modularity*.

2.1.10 Constructor dan Destructor

Constructor dan *Destructor* adalah 2 method yang akan dijalankan secara otomatis. Perbedaannya, *Constructor* baru akan dipanggil ketika Objek baru saja dibuat, sedangkan *Destructor* baru akan dijalankan ketika *Object* selesai di jalankan.

Constructor biasa digunakan sebagai proses awal yang akan selalu dijalankan, seperti koneksi ke database, sedangkan *Destructor* bisa anda gunakan untuk memutus koneksi tersebut atau hal lainnya, yakni ketika Objek selesai di jalankan.

Contoh *syntax* :

```
<?php
class Contoh{

    public function __construct(){
        echo "<p>Jalankan Koneksi ke Database</p>";
    }
    public function jalan(){
        echo "Jalankan Program";
    }
    public function __destruct(){
        echo "<p>Hentikan Koneksi ke Database</p>";
    }
}

$Program = new Contoh;
?>

<p><?php echo $Program->jalan() ?></p>
```

2.1.11 *Interface*

Dalam pemrograman berbasis objek, *interface* adalah sebuah *class* yang semua *method*-nya adalah *abstract method*. Karena semua *method*-nya adalah *abstract method* maka *interface* pun harus diimplementasikan oleh *child class* seperti halnya pada *abstract class*. Hanya saja bila kita sebelumnya menggunakan *keyword extends* untuk mengimplementasikan sebuah *abstract class*, maka pada *interface* kita menggunakan *keyword implements* untuk mengimplementasikan sebuah *interface*.

Di era *milenial* seperti sekarang ini penggunaan *interface* sangat masif. Banyak *framework* dan *library* yang kalau kita mau membaca *source code*-nya maka akan mudah sekali bagi kita untuk menemukan *interface*. Penggunaan *interface* tidak

lain karena fitur yang dimiliki *interface* itu sendiri yaitu sebagai *hirarki* tertinggi pada *parameter casting* (akan dibahas pada bab tersendiri) dimana setiap *object* yang mengimplementasikan sebuah *interface* akan *valid* jika dimasukkan kedalam *method* yang menggunakan *interface* tersebut sebagai *type hinting* atau *parameter casting*. Seperti pada *framework Laravel*, dimana *interface* akan sangat mudah ditemukan pada folder *Contracts* seperti nampak pada *Github repository Laravel* berikut. Pada paradigma pemrograman modern, ada istilah "*interface as contract*" yang maksudnya adalah *interface* digunakan pada *parameter casting* sebagai pengikat bahwa *object* yang akan *XV. Interface* 116 dimasukkan kedalam *method* pasti memiliki fitur-fitur atau *methodmethod* yang didefinisikan pada *interface* tersebut. Sehingga dengan menggunakan *interface* tersebut sebagai *parameter casting* pada *method* maka didalam *method* tersebut kita bisa dengan percaya diri untuk menggunakan *method-method* yang ada pada *interface* tanpa takut terjadi *error undefined method* .

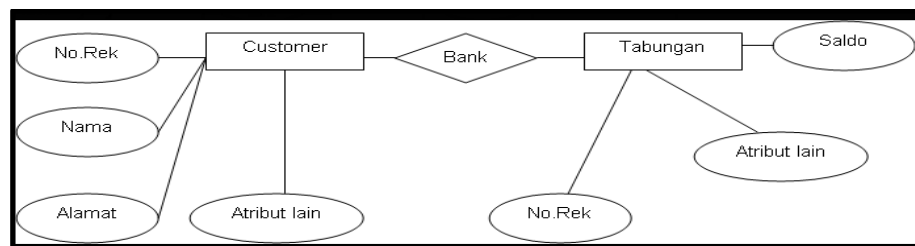
PRAKTIKUM 3

3.1.1 Model data berbasis *object*

Model data berbasis obyek ini adalah model data yang menyiapkan setiap node / chartnya dengan basis objek database. Dengan menggunakan konsep seperti *entitas*, *attribute* dan *relasi*, objek yang dimaksud adalah sebuah *entitas*.

a. Model data *semantic*

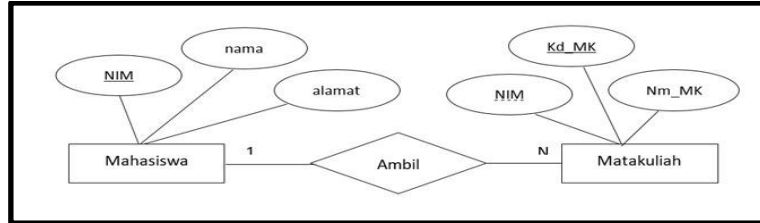
Model data *semantic* adalah relasi antar obyek yg dinyatakan dengan kata kata (*semantic*).



Gambar 3.1 1 *Semantic model*

b. Model data *ERD (Entity Relationship Diagram)*.

ERD adalah salah satu model data berbasis objek yang paling sering digunakan. Jenis dan bentuk *ERD* dari tahun ke tahun pun berbeda beda. *ERD* adalah cara penggambaran *real case* yang terjadi sesuai kasusnya. Dengan *ERD* kita bisa menggambarkan bagaimana *entitas* satu bisa terhubung dengan *entitas* lainnya.



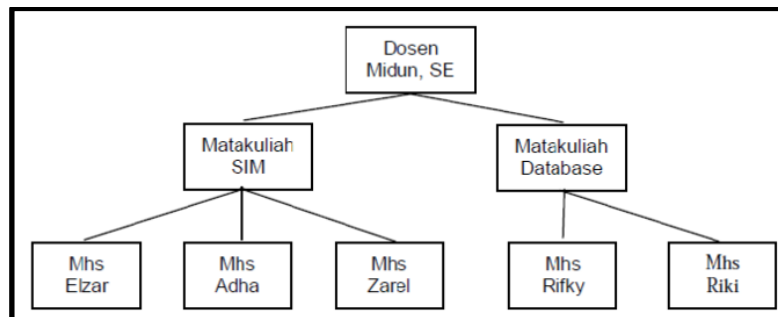
Gambar 3.1 2 ERD

3.1.2 Model data berbasis *record*

Model data ini berbeda dari model data berbasis objek. Model data ini mengambil nodenya berdasarkan record-record yang di perlukan dari database. Record sendiri adalah rekaman-rekaman data yang tersimpan di database. Contoh-contoh model data berbasis record yaitu :

- a. Model *database* hirarki.

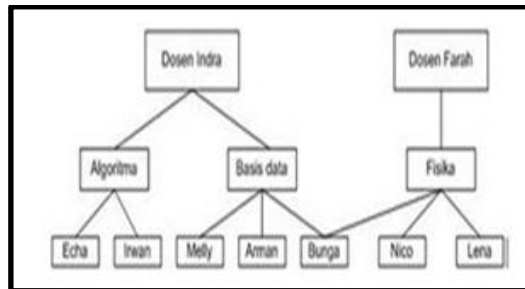
Model data ini disajikan dari kumpulan *record* dan relasi yang digambarkan seperti bentuk pohon (*tree*). Model data ini memungkinkan satu *node* hanya untuk memiliki satu orang tua.



Gambar 3.2 1 Model hirarki

b. Model *database* jaringan.

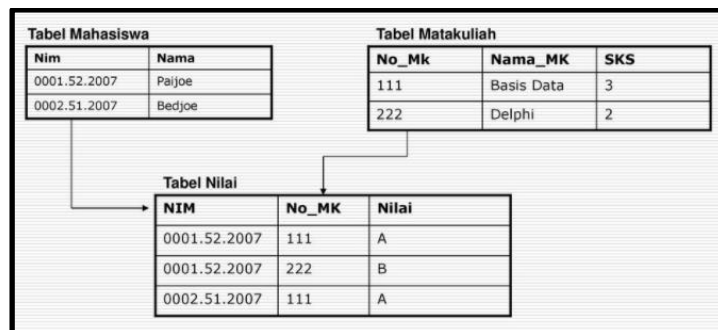
Network data model hampir menyerupai model data hirarki. Namun pada model data ini, memungkinkan satu node bisa memiliki lebih dari satu orang tua.



Gambar 3.2 2 Model jaringan

c. Model *database relational*.

Model *database* yang disajikan dalam bentuk tabel yang terdiri dari kolom dengan nama yang unik dan baris-baris yang menyimpan data yang berbeda. Model data ini digambarkan berdasarkan *recordnya* dan yang paling sering digunakan untuk memudahkan perancangan sebuah database.



Gambar 3.2 3 Model *relational*

3.1.3 Penjelasan crud

CRUD adalah singkatan yang berasal dari *Create*, *Read*, *Update*, dan *Delete*, dimana keempat istilah tersebut merupakan fungsi utama yang nantinya diimplementasikan ke dalam basis data.

Empat poin tersebut mengindikasikan bahwa fungsi utama melekat pada penggunaan *database* relasional beserta aplikasi yang mengelolanya, seperti *Oracle*, *MySQL*, *SQL Server*, dan lain – lain.

Jika dihubungkan dengan tampilan antarmuka (*interface*), maka peran CRUD sebagai fasilitator berkaitan dengan tampilan pencarian dan perubahan informasi dalam bentuk formulir, tabel, atau laporan. Nantinya, akan ditampilkan dalam *browser* atau aplikasi pada perangkat komputer *user*.

a. *Create*

Fungsi CRUD yang pertama adalah *create*, dimana anda dapat memungkinkan untuk membuat *record* baru pada sistem basis data. Jika anda sering menggunakan SQL, maka sering disebut dengan istilah *insert*.

Sederhananya, anda dapat membuat tabel atau data baru sesuai atribut dengan memanggil fungsi *create*. Akan tetapi, biasanya hanya posisi *administrator* saja yang dapat menambahkan atribut lain ke dalam tabel itu sendiri.

Tambah Kamar

Tipe Kamar :

Fasilitas :

Sistem Pembayaran :

Harga :

Masa Kontrak :

Jumlah :

Tambah Data Kos

Gambar 3.3 1 Create

b. Read

Fungsi yang kedua adalah *read*, berarti memungkinkan anda untuk mencari atau mengambil data tertentu yang berada di dalam tabel dengan membaca nilainya. Fungsi *read* mempunyai kesamaan dengan fungsi *search* yang biasa anda temukan dalam berbagai perangkat lunak.

Hal yang perlu anda lakukan adalah dengan menggunakan kata kunci (*keyword*) untuk dapat menemukan file *record* dengan bantuan *filter data* berdasarkan kriteria tertentu.

Kamar Type A	
Fasilitas	ac + televisi
Sistem Pembayaran	cash
Harga	3000000
Masa Kontrak	6 Bulan
Kamar Tersedia	2
Ubah	
Penyewa	

Gambar 3.3 2 Read

c. *Update*

Fungsi CRUD yang ketiga adalah *update*, dimana berfungsi untuk memodifikasi data atau *record* yang telah tersimpan di dalam *database*. Namun, anda perlu untuk mengubah beberapa informasi terlebih dahulu agar dapat mengubah *record* sesuai kebutuhan anda.

Untuk pengisian *update data* anda juga perlu menyesuaikan nilai atribut sesuai dengan *form* yang tersedia agar tidak ada kesalahan saat pemrosesan data di dalam *server*.



Ubah Kamar	
Tipe Kamar	: Type A
Fasilitas	: ac + televisi
Sistem Pembayaran	: cash
Harga	: 3000000
Masa Kontrak	: 6 Bulan
Jumlah	: 3
<button>Ubah Data Kamar</button>	

Gambar 3.3 3 Update

d. *Delete*

Fungsi yang terakhir adalah *delete*, dimana ketika anda tidak membutuhkan sebuah *record* lagi, maka data tersebut perlu untuk dihapus. Sehingga, anda perlu untuk menggunakan fungsi *delete* untuk memproses aktivitas tersebut.

Beberapa *software* terkait *database* relasional mengizinkan anda untuk menggunakan *soft* dan *hard delete*. Untuk *soft delete* berfungsi untuk memperbarui status baris yang menunjukkan bahwa data akan dihapus meskipun informasi

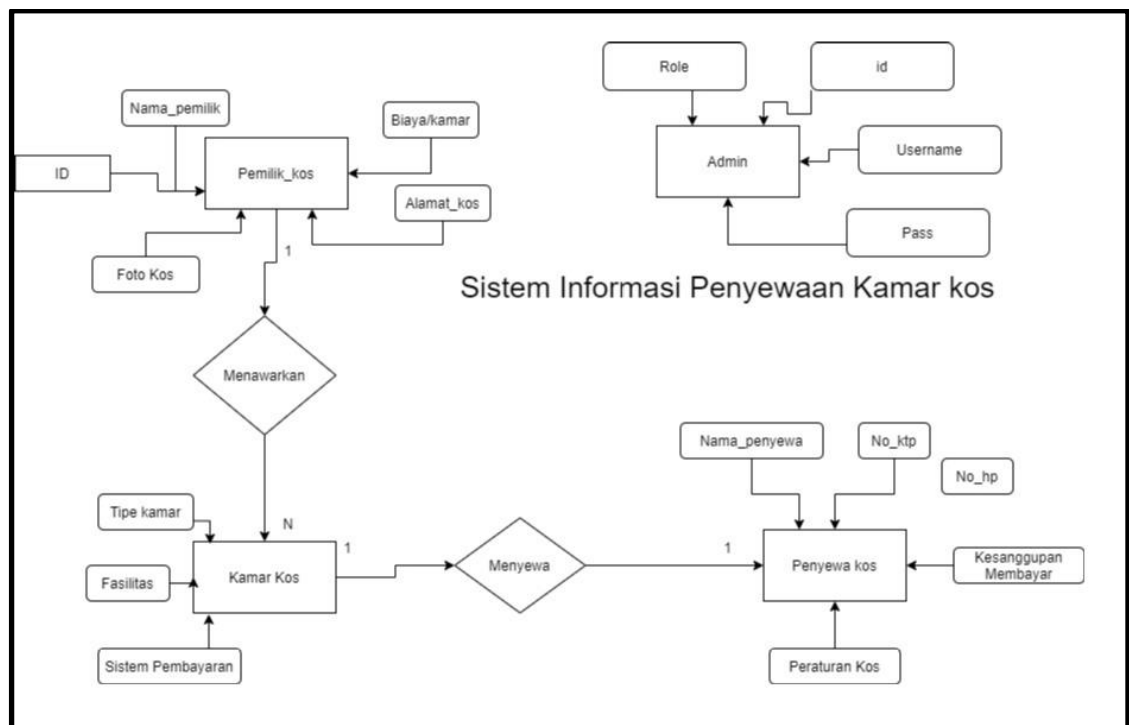
tersebut tetap ada. Sedangkan, untuk *hard delete* bertujuan untuk menghapus catatan pada basis data secara permanen.

PRAKTIKUM 4

4.1.1 Project Membuat Sistem Penyewaan Kamar Kos

1. ERD system penyewaan kamar kos

Pada pembuatan system penyewaan kamar kos ini, pertama kita membuat terlebih dahulu model data berbasis objek nya dimana model data berbasis objek tersebut berupa erd yang dimana fungsi dari erd tersebut yaitu Menjelaskan hubungan - hubungan antar data - data dalam basis data berdasarkan objek objek dasar data yang memiliki hubungan yang dihubungkan oleh suatu relasi.



Gambar 4.1 1 Entity Relationship Diagram Sistem Penyewaan Kamar kos

Gambar 4.1 menjelaskan bahwa pada erd system penyewaan kamar kos terdiri dari table pemilik kos, kamar kos, penyewa kos dan admin. Pada pemilik kos berelasi dengan kamar kos dan mempunyai hubungan relasinya *one to many* yang artinya satu pemiilik kos bias mempunyai banyak kamar kos sementara hubungan antara kamar kos dan penyewa kos berelasi dan mempunyai hubungan *one to one* artinya satu kamar kos bisa ditempati oleh satu penyewa kos, dan terakhir tabel admin dia berdiri sendiri dia berfungsi sebagai tempat untuk menyimpan user, id maupun password

4.1.2 DFD (*Data Flow Diagram*)

Data Flow Diagram (DFD) adalah suatu diagram yang menggunakan notasi notasi untuk menggambarkan arus dari data sistem, yang penggunaannya sangat membantu untuk memahami sistem secara logika, tersruktur dan jelas.

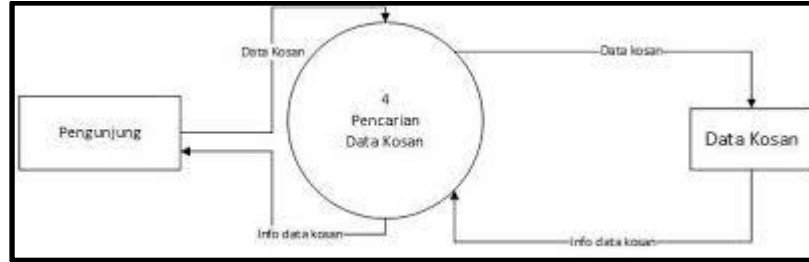
DFD merupakan alat bantu dalam menggambarkan atau menjelaskan sistem yangsedang berjalan logis. Dalam sumber lain dikatakan bahwa DFD ini merupakan salahsatu alat pembuatan model yang sering digunakan, khususnya bila fungsi-fungsisistem merupakan bagian yang lebih penting dan kompleks dari pada data yangdimanipulasi oleh sistem. Dengan kata lain, DFD adalah alat pembuatan model yangmemberikan penekanan hanya pada fungsi sistem. DFD ini merupakan alatperancangan sistem yang berorientasi pada alur data dengan konsep dekomposisidapat digunakan untuk penggambaran analisa maupun rancangan sistem yangmudah dikomunikasikan oleh profesional sistem kepada pemakai maupun

pembuatprogram. Suatu yang lazim bahwa ketika menggambarkan sebuah sistem kontekstual data flow diagram yang akan pertama kali muncul adalah interaksi antara sistem dan entitas luar. DFD didisain untuk menunjukkan sebuah sistem yang terbagi-bagi menjadi suatu bagian sub-sistem yang lebih kecil dan untuk menggarisbawahi arus data antara kedua hal yang tersebut diatas. Diagram ini lalu "dikembangkan" untuk melihat lebih rinci sehingga dapat terlihat model-model yang terdapat di dalamnya. merupakan alat yang digunakan untuk menggambarkan suatu sistem yang telah ada atau sistem baru yang akan dikembangkan secara logika tanpa mempertimbangkan lingkungan fisik dimana data tersebut mengalir ataupun lingkungan fisik dimana data tersebut akan disimpan.

a. Data flow diagram level 0

Diagram konteks atau level 0 merupakan diagram dengan tingkatan paling rendah, dimana menggambarkan sistem berinteraksi dengan entitas *eksternal*. Pada diagram konteks akan diberi nomor untuk setiap proses yang berjalan, dimulai dari angka 0 terlebih dahulu.

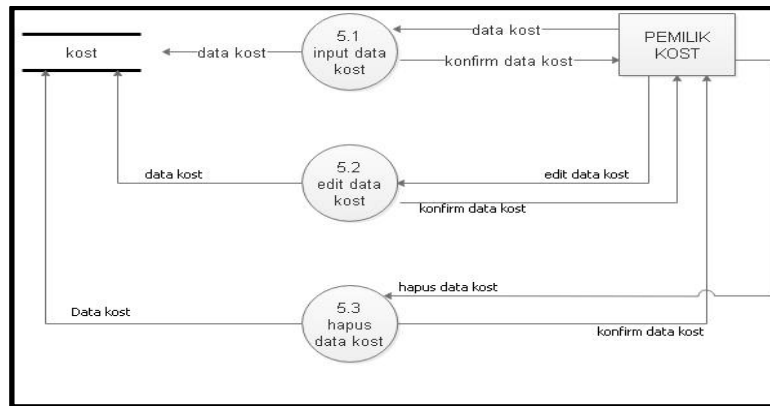
Jadi, untuk setiap aliran data akan langsung diarahkan menuju sistem. Dan ciri dari diagram level 0 terletak pada tidak adanya informasi yang terkait data yang tersimpan pada *data store*.



Gambar 4.2 1 Diagrama Flow level 0

b. Data flow diagram level 1

DFD level 1 merupakan lanjutan dari diagram konteks, dimana setiap proses yang berjalan akan diperinci pada tingkatan ini. Sehingga, proses utama akan dipecah menjadi sub – sub proses yang lebih kecil lagi.



Gambar 4.2 2 Diagram flow level 1

4.1.3 Interface

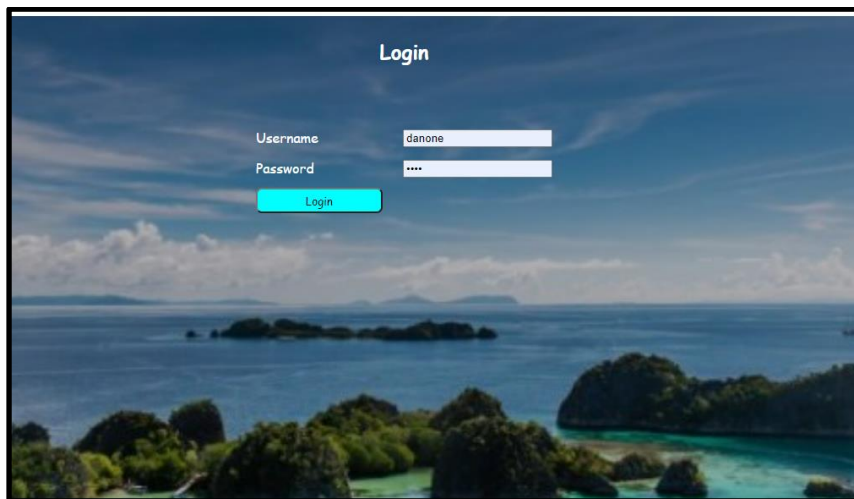
Antarmuka (*Interface*) merupakan mekanisme komunikasi antara pengguna (*user*) dengan sistem. Antarmuka (*Interface*) dapat menerima informasi dari pengguna (*user*) dan memberikan informasi kepada pengguna (*user*) untuk membantu mengarahkan alur penelusuran masalah sampai ditemukan suatu solusi

Interface, berfungsi untuk menginput pengetahuan baru ke dalam basis pengetahuan sistem pakar (ES), menampilkan penjelasan sistem dan memberikan panduan pemakaian sistem secara menyeluruh / *step by step* sehingga pengguna mengerti apa yang akan dilakukan terhadap suatu sistem. Yang terpenting adalah kemudahan dalam memakai / menjalankan sistem, interaktif, komunikatif, sedangkan kesulitan dalam mengembangkan / membangun suatu program jangan terlalu diperlihatkan.

Interface yang ada untuk berbagai sistem, dan menyediakan cara : *Input*, memungkinkan pengguna untuk memanipulasi sistem. *Output*, memungkinkan sistem untuk menunjukkan efek manipulasi pengguna.

a. *Login*

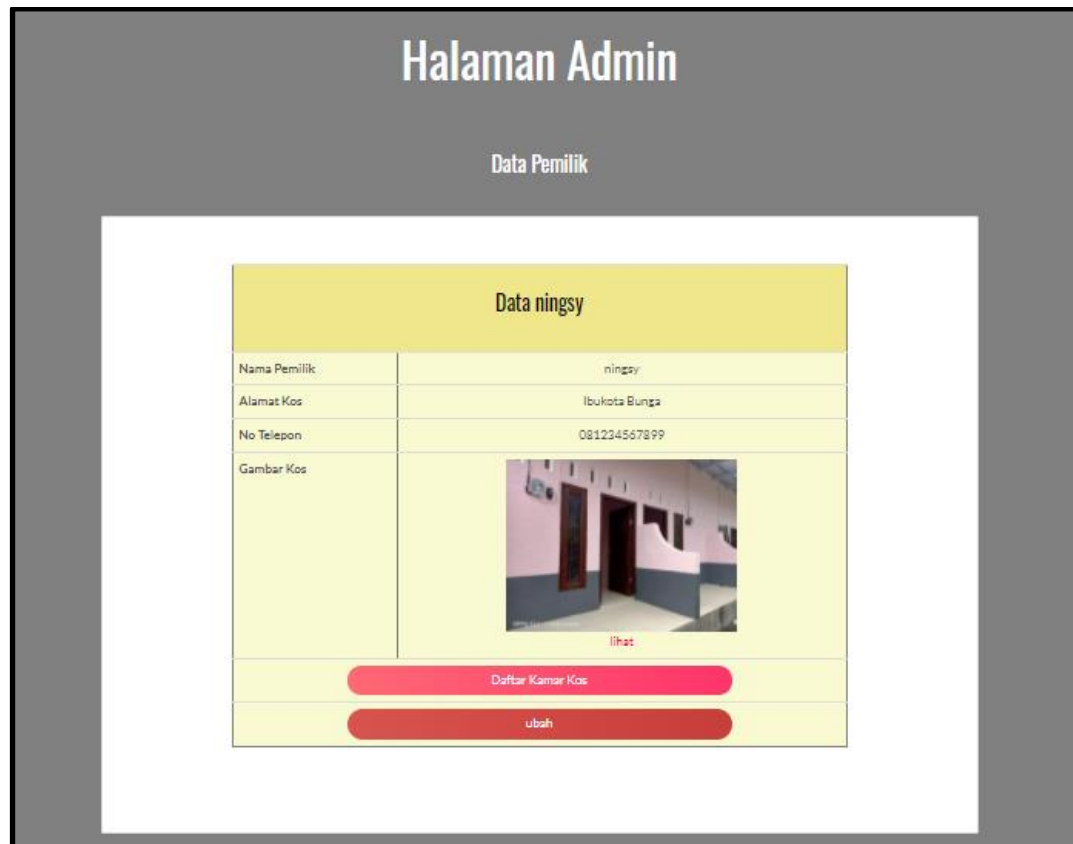
Ketika kita mengakses halaman ini kita akan diarahkan untuk login terlebih dahulu sebelum memasuki halaman selanjutnya



Gambar 4.3 1 *Login*

b. Halaman Admin

Setelah admin login, maka kita akan diarahkan kehalaman selanjutnya yaitu halaman admin



Gambar 4.3 2 Admin

Pada tampilan halaman admin terdapat beberapa data pemilik kos mulai dari nama pemilik, alamat kos, no telpon serta gambar kos nya. Pada halaman admin juga terdapat 2 tombol yaitu tombol daftar kamar kos dan tombol ubah

SEMOGA BETAH YAH!!!

Tambah Kamar

Kamar Type A	
Fasilitas	ac + televisi
Sistem Pembayaran	cash
Harga	3000000
Masa Kontrak	6 Bulan
Kamar Tersedia	2
Ubah	
Penyewa	

Gambar 4.3 3 Daftar kamar kos

Tambah Kamar

Tipe Kamar :

Fasilitas :

Sistem Pembayaran :

Harga :

Masa Kontrak :

Jumlah :

Tambah Data Kos

Gambar 4.3 4 Tambah Data kos

Dan ketika tombol tambah data kos diklik maka data tersebut akan ditambahkan dan akan tersimpan

c. Penyewaan kamar kos

Pada halaman ini terdapat informasi mengenai kamar kos yang ingin kita pilih mulai dari fasilitas sampai kamar yang tersedia. Pada halaman ini juga terdapat tombol informasi pemilik kos dan tombol untuk pilih untuk memilih kos tersebut

PENYEWAAN KAMAR KOS

Login

Daftar Kamar Kos

Kamar Type A	
Fasilitas	ac + televisi
Sistem Pembayaran	cash
Harga	3000000
Masa Kontrak	6 Bulan
Kamar Tersedia	2
Informasi Pemilik	
Pilih	

Gambar 4.3 5 Penyewaan Kamar Kos

Untuk Menyewa Kamar Kos Silahkan Isi data diri Anda

Nama Penyewa:

No KTP:

No Telepon:

Kesanggupan Membayar:

Masa Kontrak:

6 Bulan Rp 3000000

Sewa Kamar Ini

Setelah Mengklik "Sewa Kamar Ini" silahkan lakukan pembayaran pada pemilik kos sebesar Rp 3000000
Kunci kamar kos akan diberikan oleh pemilik_kos setelah anda menyelesaikan proses pembayaran.

Untuk melihat informasi pemilik kos silahkan klik link dibawah

informasi pemilik kos

Kembali

Gambar 4.3 6 Penyewaan Kamar kos

Setelah mengisi data penyewaan maka kita akan di rahkan kehalam penyewa

Kamar Type A					
PENYEWA					
Nama Penyewa	No KTP	No Telepon	Kesanggupan Membayar	Status	UPDL
Shirahosh	9193627491736473	082147833244	Sanggup	Sudah Bayar	Ubah Hasil
Vegapunk	3647284664748263	082257748009	Sanggup	Sudah Bayar	Ubah Hasil
Rizki	1368	0890	sanggup	Konfirmasi Pembayaran	Ubah Hasil

Gambar 4.3 7 Halaman Penyewa

Pada halaman ini terdapat informasi mengenai status pembayaran kamar kos apa kita membayara maka akan tertulis sudah membayar dan ketika jika belum membayar akan diminta kita untuk melakukan pembayaran yang tulisannya berupa konfirmasi pembayaran

DAFTAR PUSTAKA

Huda, Nurul . 2020. *Pengertian Composer dan Cara Menggunakannya*.

Bandung

Hadi, Diki Alfarabi. 2021. *Pengertian Class, Object, Property, Dan Method*.

Jakarta

Wibowo, Kadek .2015.*Pengertian Pemograman Berorientasi Objek*.

Yogyakarta

Syarief, Alfa Farhan. 2021. *Database Instrumental 3 – Model Data*. Bekasi

Adani, Muhammad Robith. 2020. *Fungsi Dan Kelebihan Dari Penggunaan*

CRUD Dalam Pemrograman. Jakarta