

**LAPORAN LENGKAP
PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK**



OLEH :

NAMA : LM.HAFIDZ ABDILLAH.SM
NIM : F1G120024
KELOMPOK : II (DUA)

ASISTEN PENGAMPU :
WAHID SAFRI JAYANTO (F1G117059)

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

KATA PENGANTAR



Puji syukur kami panjat kankehadirat Allah SWT, karena berkat rahmat dan hidayah-nya penyusunan laporan Pemrograman beriorientasi objek dapat di selesaikan dengan tepat waktu tanpa ada halangan yang berarti.

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

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

Akhirnya saya sebagai penyusun mengharapkan semoga dari laporan praktikum PBO ini dapat diambil hikma dan maanfaatnya sehingga dapat memberikan inspirasi terhadap pembaca.

Kendari, Desember 2021

Penulis

HALAMAN PENGESAHAN

LAPORAN PRAKTIKUM



OLEH:

LM.HAFIDZ ABDILLAH.SM
F1G120024

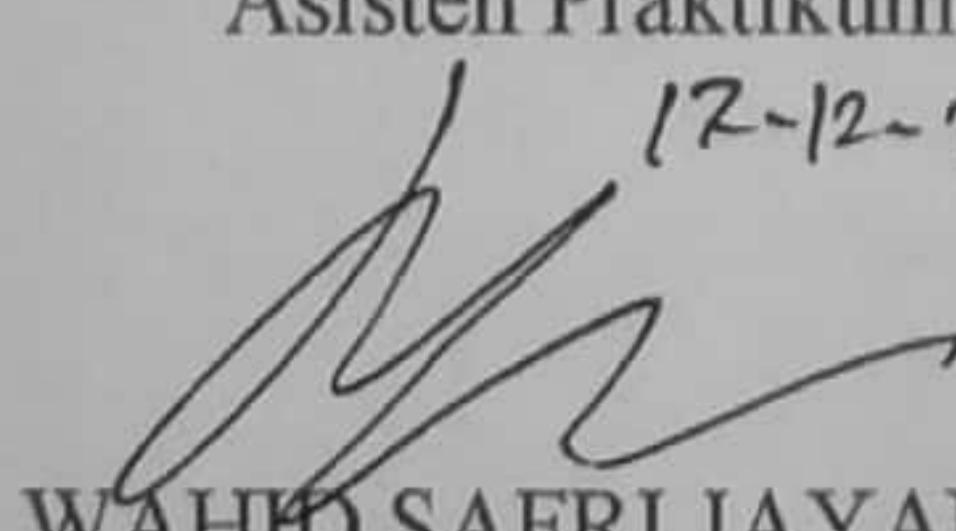
Laporan praktikum Pemrograman Berorientasi Objek ini disusun sebagai tugas akhir menyelesaikan praktikum Pemrograman Berorientasi Objek sebagai salah satu syarat lulus matakuliah Pemrograman Berorientasi Objek. Menerangkan bahwa yang tertulis dalam laporan lengkap ini adalah benar dan dinyatakan telah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

Kendari, 12 Desember 2021

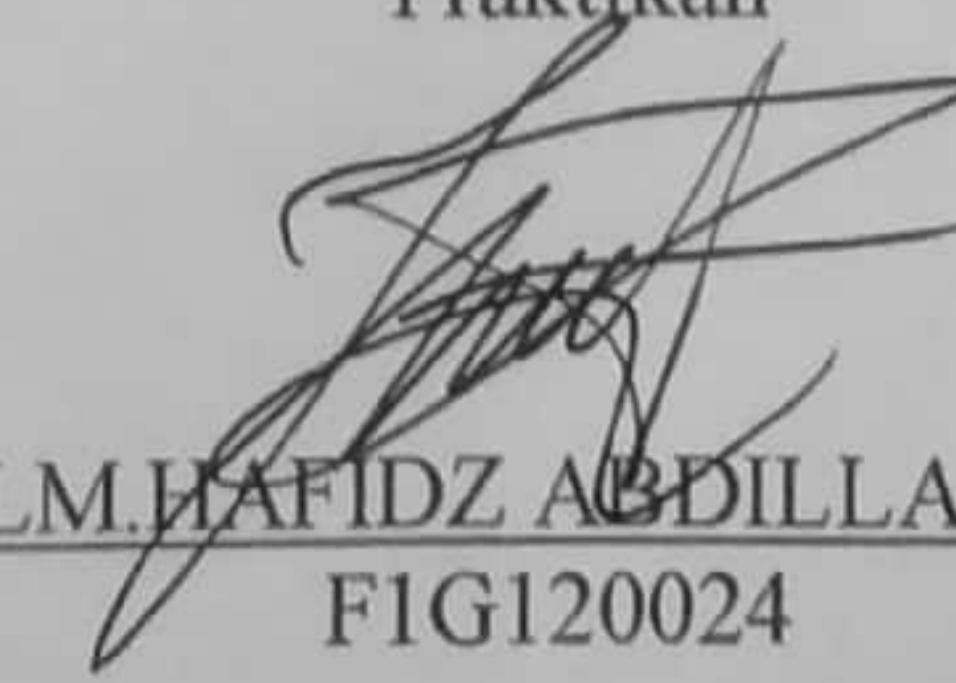
Menyetujui

Asisten Praktikum

12-12-2021


WAHID SAFRI JAYANTO
F1G117059

Praktikan


LM.HAFIDZ ABDILLAH.SM
F1G120024

DAFTAR ISI

HALAMAN COVER	i
KATA PENGANTAR.....	ii
HALAMAN PENGESAHAN.....	iii
DAFTAR ISI.....	iv
DAFTAR TABEL	vi
DAFTAR GAMBAR.....	vii
PERTEMUAN PERTAMA.....	1
1.1. Alat dan Bahan	1
1.2. Pengenalan PBO (Pemrograman Berorientasi Objek)	1
1.3. Pengenalan <i>PHP</i>	4
PERTEMUAN KE DUA.....	7
2.1. <i>Class</i>	7
2.2. <i>Method</i>	7
2.3. <i>Constructor</i>	8
2.4. <i>Property</i>	9
2.5. <i>Object</i>	9
2.6. <i>Modifier</i>	10
2.7. <i>Composer</i>	10
2.8. Laravel.....	11
2.9. <i>Constructor dan Destructor</i>	12
2.9. <i>Interface</i>	13
PERTEMUAN KE TIGA	15
3.1. Pengertian <i>CRUD</i>	15
3.2. Fungsi <i>CRUD</i>	15

3.2. Project pertama membuat <i>crud member</i> dan golongan.....	16
PERTEMUAN KE EMPAT.....	19
4.1. <i>ERD</i>	19
4.2. <i>DFD</i>	21
4.3. <i>Interface / Tampilan antar muka</i>	24
1. Halaman Login.....	25
2. Halaman Admin.....	26
3. Halaman Penyewaan	26
4.4. Kesimpulan.....	27
4.5. Saran.....	28
DAFTAR PUSTAKA	29

DAFTAR TABEL

Halaman

Tabel 1. 1 Alat dan Bahan.....	1
--------------------------------	---

DAFTAR GAMBAR

	Halaman
Gambar 3.1 1 Halaman Login.....	16
Gambar 3.1 2 Halaman <i>Member</i>	17
Gambar 3.1 3 Halaman Manager	18
Gambar 4. 1 <i>ERD</i> sistem penyewaan kos	20
Gambar 4. 2 Diagram konteks	22
Gambar 4. 3 Diagram <i>flow level 1</i>	23
Gambar 4. 4 Halaman <i>Login</i>	25
Gambar 4. 5 Halaman Admin	26
Gambar 4. 6 Halaman Penyewaaan kos.....	27

PERTEMUAN PERTAMA

1.1. Alat dan Bahan

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

NO	ALAT DAN BAHAN	KEGUNAAN
1	Labtop	perangkat keras (<i>hardware</i>) yang digunakan untuk menyimpan aplikasi
2	Xampp	XAMPP digunakan untuk membuat <i>web</i> server lokal pada komputer
3	Visual Studio Code	Digunakan sebagai tempat untuk membuat <i>codingan</i>
4	Phpmyadmin	Digunakan untuk membuat <i>database</i>
5	Chrome	Digunakan untuk menampilkan sebuah project yang kita buat

Tabel 1. 1 Alat dan Bahan

1.2. Pengenalan PBO (Pemrograman Berorientasi Objek)

Secara garis besar, bahasa pemrograman komputer adalah sebuah alat yang dipakai oleh para programmer komputer untuk menciptakan program aplikasi yang digunakan untuk berbagai macam keperluan. Pada tahap awal dikenal beberapa jenis bahasa pemrograman, bahasa ini berbasis teks dan berorientasi linear contohnya: Bahasa BASIC, Bahasa Clipper, Bahasa Pascal, Bahasa cobol.

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 rupaan. 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 meyimpan *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, di mana di dalam *body* kelas ini didefinisikan *variable*, *method*, dan kelas *inner*. Deklarasi kelas automatis 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 dan 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 dibagi menjadi beberapa kategori yaitu:

a. 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.

b. Fungsi / 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 dijalankan atau dieksekusi.

1.3. 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 perilisan 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 pemograman *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*

Penulisan *syntax PHP*, *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 *semicolon* atau titik koma (;) berikut ini adalah contoh penulisan *syntax php* yang benar

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

PERTEMUAN KE DUA

2.1. *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  
  
Class Nama_class{  
    //isi dari class  
}  
?>
```

2.2. *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. Contohnya sebagai berikut :

```
<?php  
  
class Pesawat{  
    var $Warna;  
    var $Jumlah_Penumpang;  
    var $Maskapai;  
    function Terbang() { //isi method}  
    Function Mendarat() { //isi method}  
}  
?>
```

2.3. 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* sebagai berikut :

```
<?php  
class Siswa {  
    public method __construct(){  
        echo "Fungsi Construct Terpanggil";  
    }  
    Public function __destruct() {  
        echo "Fungsi Destruct Terpanggil";  
    }  
}  
// function __construct() terpanggil Ketika objek ini dibuat  
$Hafidz = new siswa ();  
Echo "<br>Batas<br/>";  
// function __destruct () terpanggil Ketika file berakhir  
?>
```

2.4. *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*.

Contoh *syntax property*

```
<?php  
class Motor{  
    var $warna;  
    var $merek;  
    var $jenis; }  
?>
```

2.5. *Object*

Object adalah hasil konkret atau hasil cetakan dari sebuah *class*. Sebagai misalnya saya telah membuat *class user* maka *objectnya* adalah para *user* atau *account*, misalnya Hafidz, Aqso dan Aulia.

Contoh *syntax object*

```
<?php  
class orang{  
    //isi class  
}  
$Aqso = new orang();  
?>
```

2.6. *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. Contoh *syntax modifier* :

```
Public class bank balance
{
    public String owner
    public int balance

    public bank_balance(String name, int dollars )
    {
        owner = name;

        if(dollars > = 0)
            balance = dollars;
        else
            dollars =0;
    }
}
```

2.7. *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 Contoh penggunaan *composer* :

```
<?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  
]);
```

2.8. 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*.

2.9. *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>";
    }
}
```

2.9. 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 *method-method* yang didefinisikan pada *interface* tersebut. Sehingga dengan menggunakan *interface* tersebut sebagai paramter *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*.

PERTEMUAN KE TIGA

Pada pertemuan ke tiga ini membahas tentang apa yang dimaksud dengan crud serta fungsi dari crud itu sendiri dan juga project pertama yaitu membuat website yang menampilkan 2 tabel yang berelasi yang didalamnya terdapat fungsi atau dapat menggunakan fungsi crud sebagai fitur fitur yang dapat kita gunakan.

3.1. Pengertian *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. 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.

3.2. Fungsi *CRUD*

Berikut ini empat poin penting dari akronim fungsi CRUD untuk mengembangkan perangkat lunak, baik berbasis web maupun mobile yaitu :

- a. Fungsi CRUD yang pertama adalah create, dimana anda dapat memungkinkan untuk membuat record baru pada sistem basis data.
- b. Fungsi yang kedua adalah read, berarti memungkinkan anda untuk mencari atau mengambil data tertentu yang berada di dalam tabel dengan membaca nilainya.
- c. Fungsi CRUD yang ketiga adalah update, dimana berfungsi untuk memodifikasi data atau record yang telah tersimpan di dalam database.

d. Fungsi yang terakhir adalah delete, dimana ketika anda tidak membutuhkan sebuah record lagi, maka data tersebut perlu untuk dihapus.

3.2. Project pertama membuat *crud member* dan golongan

Menjelaskan projek saya tentang *crud member* dan golongan serta saya akan menampilkan gambar beserta keterangannya dan juga beserta erdnya

1. Halaman *login* / masuk

Pada halaman ini kita diminta untuk masuk bisa melalui member maupun manager



Gambar 3.1 1 Halaman *Login*

2. Halaman *member*

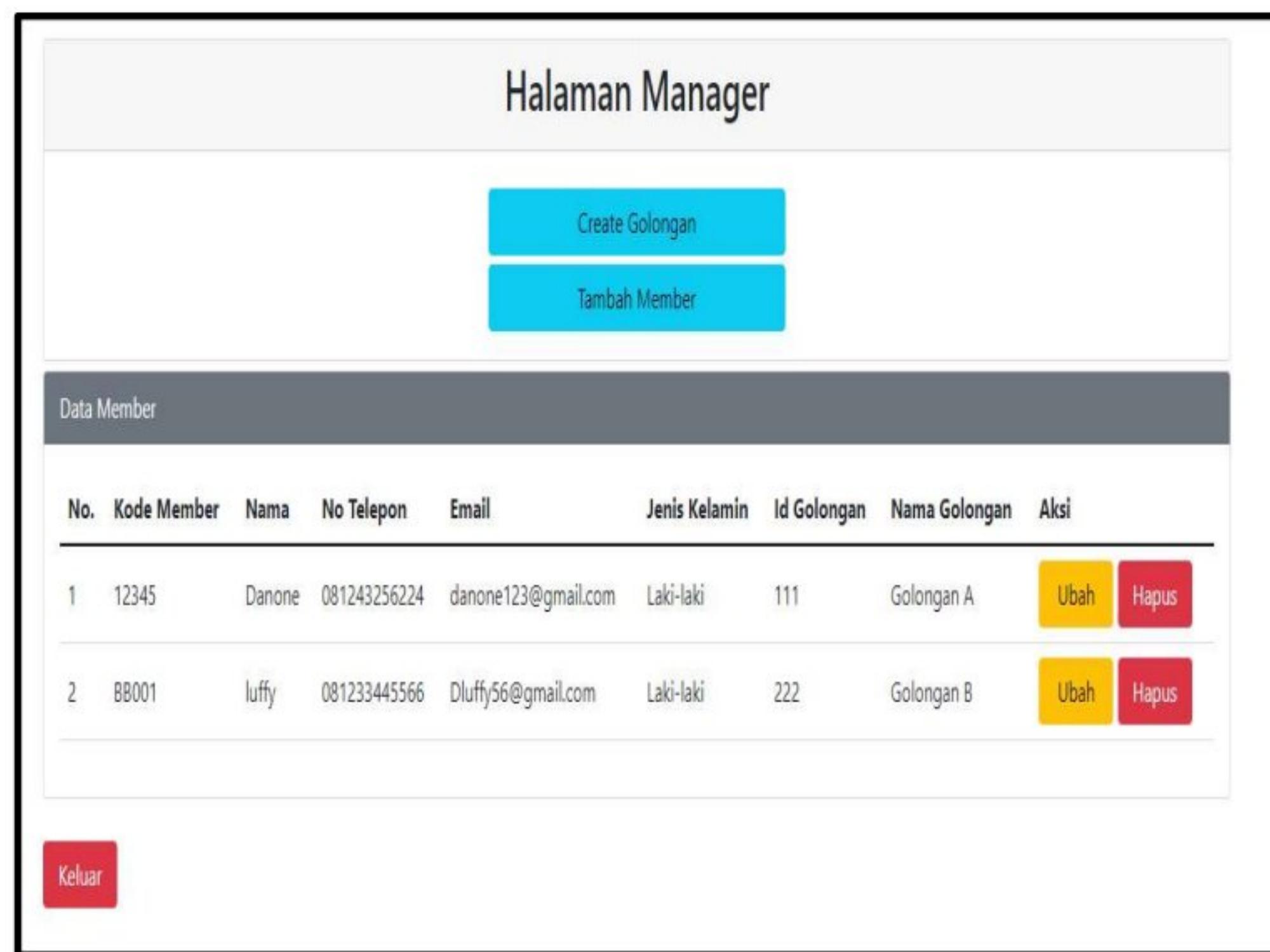
Ketika kita masuk sebagai member kita akan di arahkan ke halaman member dimana pada halaman member terdapat data data dari member itu sendiri baik dari kode *member*, nama *member* dll. Di halaman member terdapat tombol untuk mengubah data member dan *logout* untuk keluar dari halaman member itu sendiri.

Data Member	
Kode Member	: BB001
Nama	: luffy
No Telepon	: 081233445566
Email	: Dluffy56@gmail.com
Jenis Kelamin	: Laki-laki
Id Golongan	: 222
Nama Golongan	: Golongan B
Ubah Data	
Logout	

Gambar 3.1 2 Halaman *Member*

3. Halaman manager

Ketika masuk sebagai menager kita akan di arahkan ke halaman menager, pada halaman menager terdapat tombol untuk menambahkan golongan dan member. Kemudia pada halaman menager terdapat informasi mengenai kode member, nama no telpoon, email, jenis kelamin, id golongan, nama golongan dan aksi. Aksi ini dia berfungsi untuk kita dapat mengubah data di halaman menager atau pun menghapus data dihalaman menager, serta dihalaman menager juga terdapat tombol keluar yang berfungsi untuk kita dapat keluar dari halaman menager tersebut



Gambar 3.1 3 Halaman Manager

PERTEMUAN KE EMPAT

Pada pertemuan ini kami melakukan praktikum tentang projek akhir yaitu membuat aplikasi kos berbasis *website*, dan yang menjadi dasar pokok bahasan dalam kegiatan praktikum pertemuan kali ini adalah Pengolahan data pembayaran masih menggunakan sistem *konvensional* atau manual yaitu melakukan pencatatan kedalam buku catatan pembayaran. Dengan pengolahan sistem manual, kendala yang dihadapi adalah pengecekan data penyewa yang telah membayar maupun yang belum, pengecekan data kamar yang kosong, dan pencarian data penyewa. Dari permasalahan tersebut, maka dibangun Sistem Informasi berbasis *web* agar dapat digunakan untuk membantu pemilik kos mengolah berbagai administrasi dan keuangan kos. Sistem ini dapat membantu calon penyewa memonitoring kamar kos yang telah terisi, rusak, maupun yang belum terisi, dapat membantu penyewa kos dalam pembayaran kos, dan dapat membantu pemilik kos dalam membuat laporan keuangan.

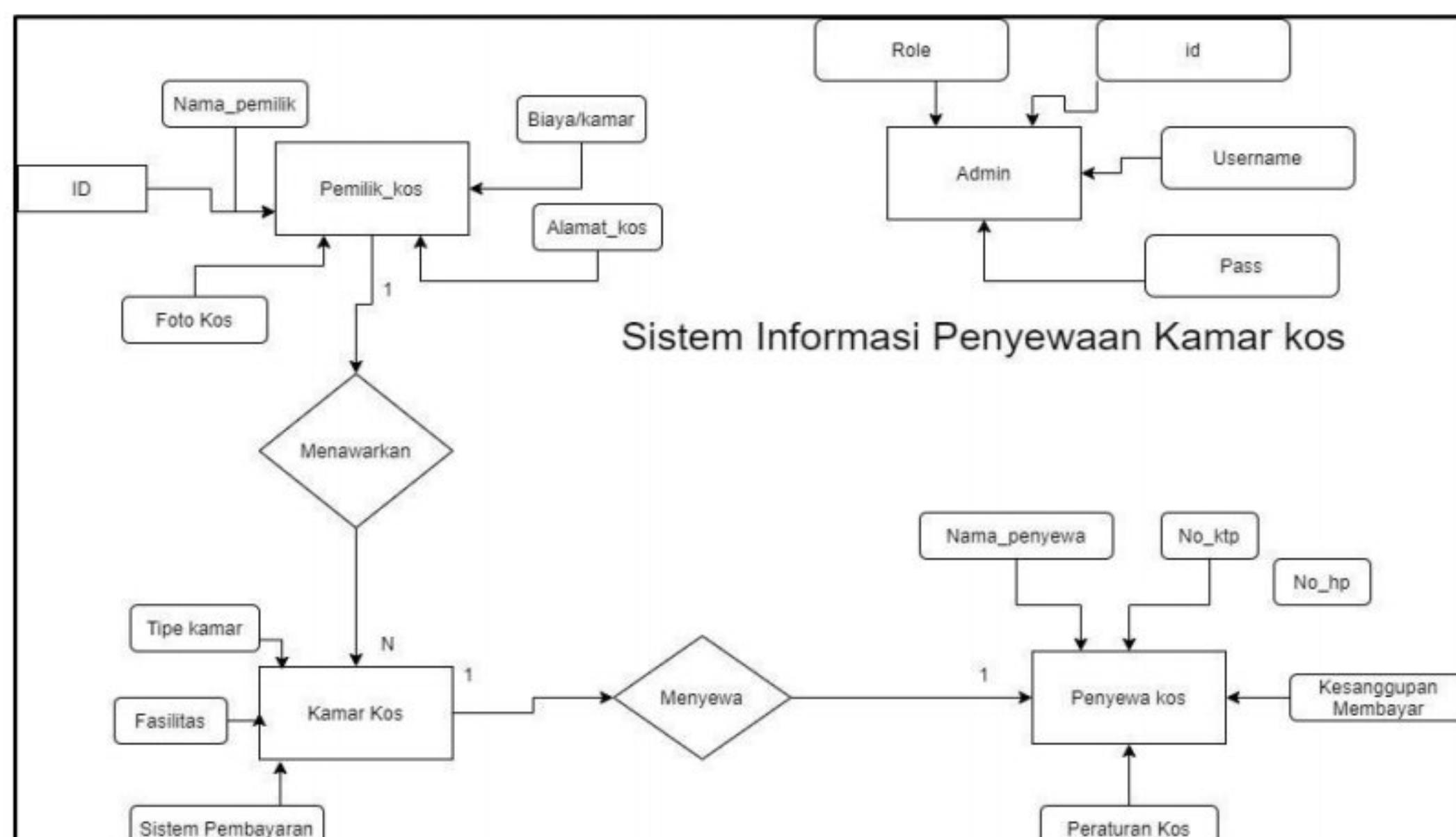
4.1. ERD

Entity Relationship Diagram (ERD) merupakan konseptual representasi data dan secara abstrak. *Entity Relationship* sendiri adalah salah satu metode pemodelan basis data yang digunakan untuk menghasilkan skema konstektual untuk jenis atau model data semantik sistem.

1) Fungsi *ERD*

Fungsi penggambaran *Entity Relationship Diagram (ERD)* yang ada saat ini yaitu sebagai berikut :

- a. Untuk memudahkan kita dalam menganalisis pada suatu basis data atau suatu sistem dengan cara yang cepat dan murah.
- b. Dapat dilakukan pengujian pada model yang telah dibuat dan dapat mengabaikan proses yang sudah dibuat hanya dengan menggambar *Entity Relationship Diagram (ERD)*.
- c. Menjelaskan hubungan-hubungan antar data-data dalam basis data berdasarkan objek –objek dasar data yang memiliki hubungan yang dihubungkan oleh suatu relasi.
- d. Untuk mendokumentasikan data-data yang ada dengan cara *mengidentifikasi* setiap *entitas* dari data-data dan hubungannya pada suatu *Entity Relationship Diagram (ERD)* itu sendiri.



Gambar 4. 1 ERD sistem penyewaan kos

Gambar 4.1 1 menjelaskan model data yang digunakan yaitu *Entity Relationship model* (ERD). Merupakan model untuk menjelaskan hubungan antar data dalam basis data berdasarkan persepsi bahwa *real world* terdiri dari objek-objek dasar yang mempunyai hubungan / relasi antara objek tersebut.

4.2. DFD

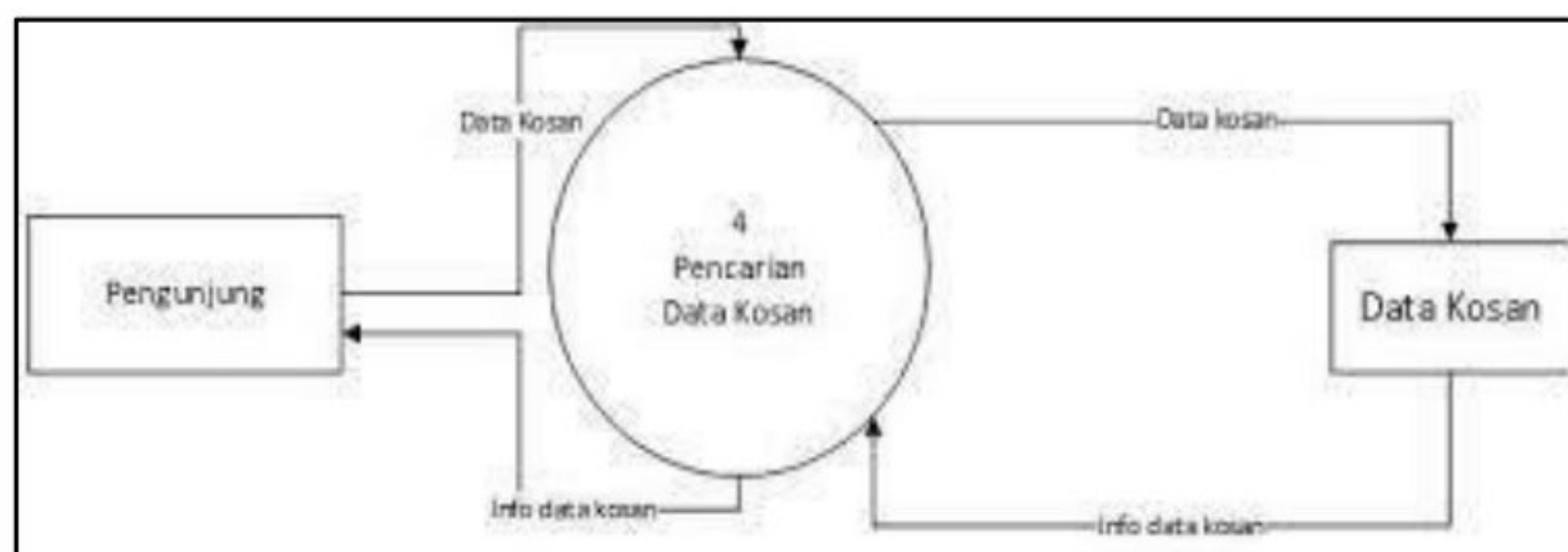
Data Flow Diagram (DFD) adalah suatu diagram yang menggunakan notasi-notasi untuk menggambarkan arus dari data sistem, yang penggunaannya sangat membantu untuk memahami sistem secara logika, tersuktur dan jelas.

DFD merupakan alat bantu dalam menggambarkan atau menjelaskan sistem yang sedang berjalan logis. Dalam sumber lain dikatakan bahwa *DFD* ini merupakan salah satu alat pembuatan model yang sering digunakan, khususnya bila fungsi-fungsi sistem merupakan bagian yang lebih penting dan kompleks dari pada data yang dimanipulasi oleh sistem. Dengan kata lain, *DFD* adalah alat pembuatan model yang memberikan penekanan hanya pada fungsi sistem. *DFD* ini merupakan alat perancangan sistem yang berorientasi pada alur data dengan konsep dekomposisi dapat digunakan untuk penggambaran analisa maupun rancangan sistem yang mudah dikomunikasikan oleh profesional sistem kepada pemakai maupun pembuat program. 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 menjadisatu

bagian sub-sistem yang lebih kecil adan untuk menggaris bawahi 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. Diagram level 0

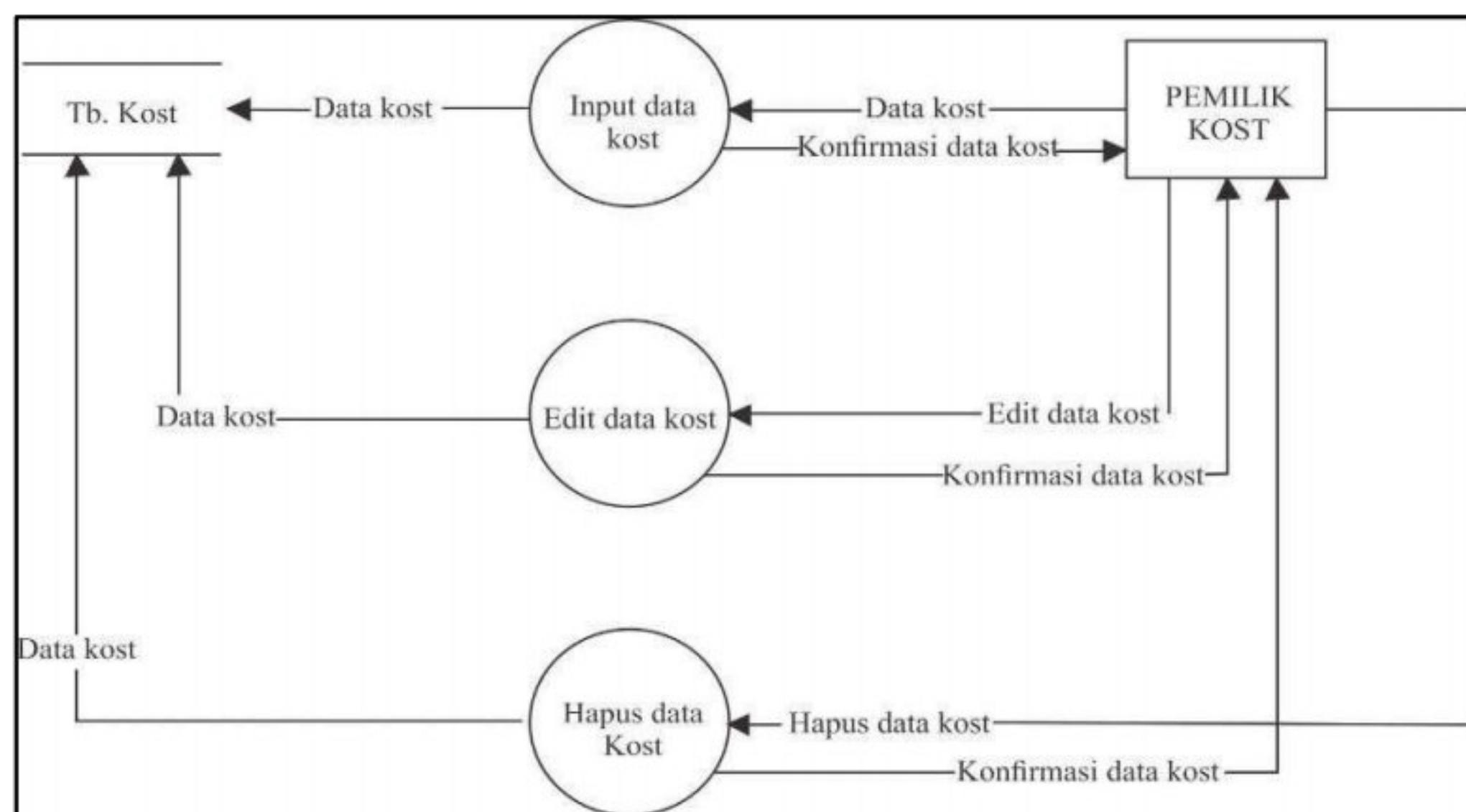
Diagram level 0 atau bisa juga diagram konteks adalah level diagram paling rendah yang menggambarkan bagaimana sistem berinteraksi dengan external entitas. Pada diagram konteks akan diberikan nomor untuk setiap proses yang berjalan, umumnya mulai dari angka 0 untuk start awal. Semua entitas yang ada pada diagram konteks termasuk juga aliran datanya akan langsung diarahkan kepada sistem. Pada diagram konteks ini juga tidak ada informasi tentang data yang tersimpan dan tampilan diagramnya tergolong sederhana.



Gambar 4. 2 Diagram konteks

b. *Data flow diagram* level 1

DFD level 1 adalah tahapan lebih lanjut tentang DFD level 0, dimana semua proses yang ada pada DFD level 0 akan dirinci dengan lengkap sehingga lebih lengkap dan detail. Proses-proses utama yang ada akan dipecah menjadi sub-proses.



Gambar 4. 3 Diagram flow level 1

Ada perbedaan antara 2 level DFD tersebut yang perlu diketahui, berikut ini perbedaannya:

1. DFD level 0 hanya mengambarkan sistem secara basic saja.
2. DFD level 0 hanya menjelaskan aliran data dari input sampai output.
3. DFD level 1 mengambarkan aliran data yang lebih kompleks pada setiap prosesnya yang kemudian terbentuklah data store dan aliran data.
4. DFD level 1 mengambarkan sistem secara sebagian atau seluruhnya secara mendetail.

4.3. *Interface* / Tampilan antar muka

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.

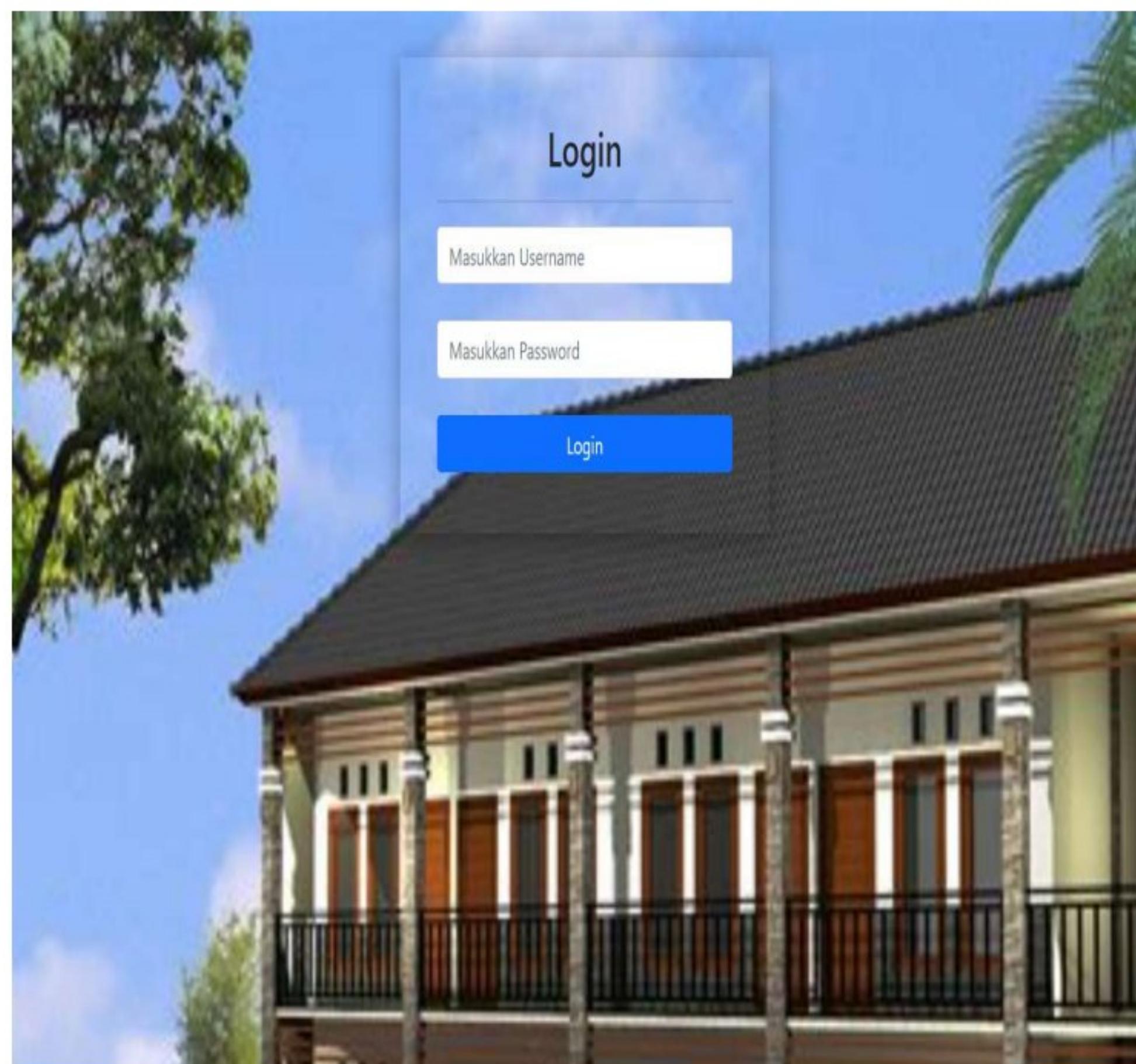
Tujuan *Interface*

Tujuan sebuah *interface* adalah mengkomunikasikan fitur-fitur sistem yang tersedia agar *user* mengerti dan dapat menggunakan sistem tersebut. Dalam hal ini penggunaan bahasa amat efektif untuk membantu pengertian, karena bahasa merupakan alat tertua (barangkali kedua tertua setelah gesture) yang dipakai orang untuk berkomunikasi sehari-harinya. Praktis, semua pengguna komputer dan Internet (kecuali mungkin anak kecil yang memakai komputer untuk belajar membaca) dapat mengerti tulisan. Meski pada

umumnya panduan *interface* menyarankan agar ikon tidak diberi tulisan supaya tetap mandiri dari bahasa, namun elemen *interface* lain seperti teks pada tombol, *caption window*, atau teks-teks singkat di sebelah kotak input dan tombol pilihan semua menggunakan bahasa. Tanpa bahasa pun kadang ikon bisa tidak jelas maknanya, sebab tidak semua lambang ikon bisa bersifat universal.

1. Halaman Login

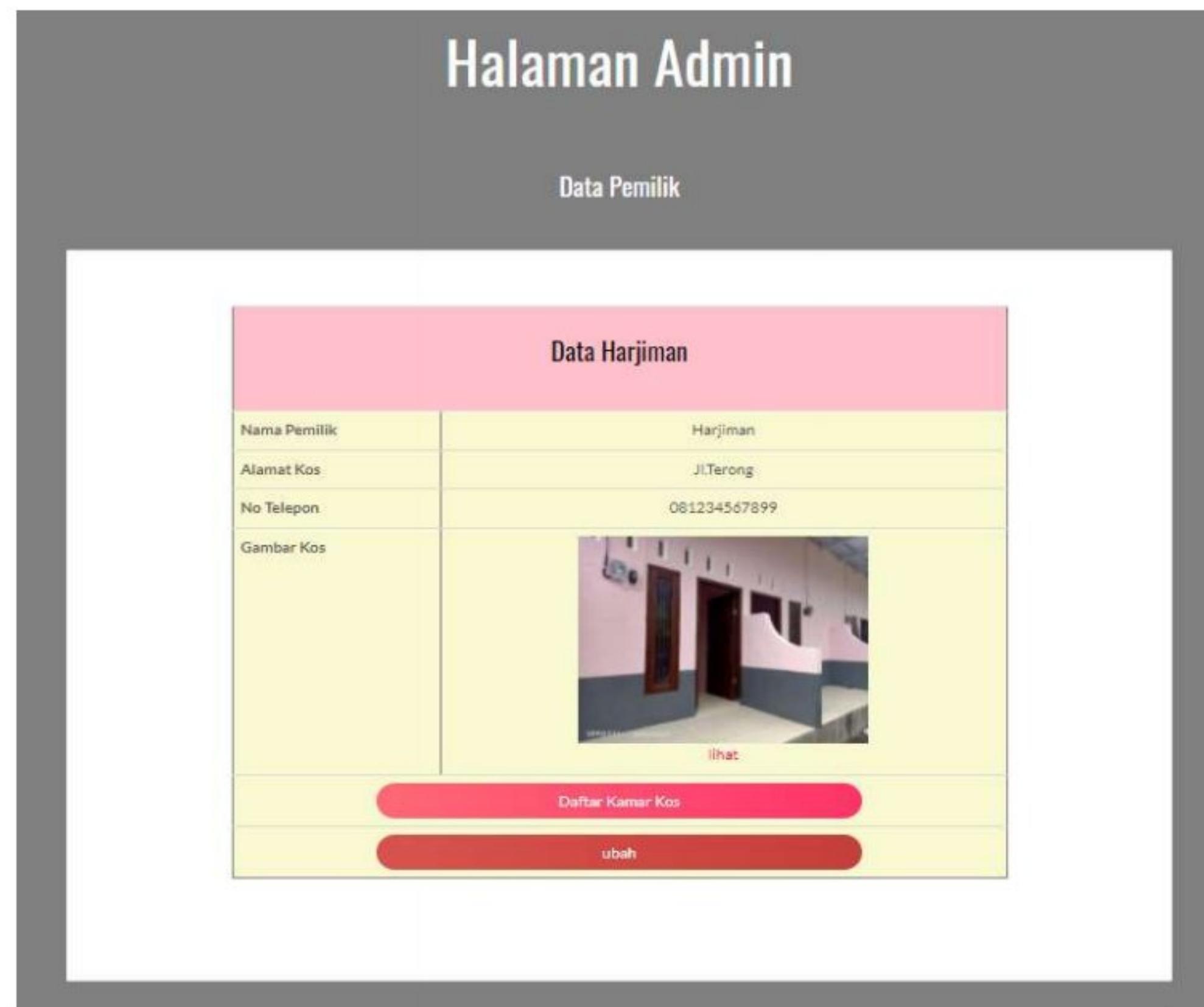
Pada halaman ini kita arahkan untuk mengisi username dan password sebelum diarahkan kehalaman selanjutnya



Gambar 4. 4 Halaman Login

2. Halaman Admin

Pada halaman ini 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.



Gambar 4. 5 Halaman Admin

3. Halaman Penyewaan

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

Permintaan Sewa Kos

Peraturan Kos

1. menghemat air dengan menutup keran setelah menggunakan air
2. menghemat listrik dengan cara mematikan lampu dan peralatan elektronik yang tidak digunakan, terutama saat akan meninggalkan kost
3. menjaga fasilitas kost dengan baik, apabila ada fasilitas atau bagian dari kamar kost yang rusak harap segera menghubungi pengelola kost
4. mengunci kamar pribadi maninggalkan kost

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 kost sebesar Rp 3000000
Kunci kamar kos akan diberikan oleh pemilik_kos setelah anda menyelesaikan proses pembayaran.

Untuk melihat informasi pemilik kost silahkan klik link dibawah

[informasi pemilik kos](#)

[Kembali](#)

Gambar 4. 6 Halaman Penyewaan kos

4.4. Kesimpulan

- a. Pemrograman Berorientasi Objek (*Object Oriented Programming* atau *OOP*) merupakan paradigma pemrograman yang berorientasikan kepada objek.
- b. *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*.

- c. *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*
- d. *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.
- e. *property* adalah data-data yang terdapat didalam *class*. datanya bisa berupa sifat. kegunaan *property* pada sebuah *class* sama dengan kegunaan variabel *diphp* bisa digunakan untuk menyimpan data dan lain lain
- f. *Object* adalah *output* dan *class* dan *object* juga dapat menampilkan atau mengelola isi *class*

4.5. Saran

Saran untuk praktikum kali ini yaitu agar lebih menekankan pemahaman terhadap materi dan praktek terhadap materi materi atau pokok bahasan yang di praktikumkan dan juga pada saat pembuatan projek agar lebih mempelajari lagi cara membuat codinngan untuk membangun sebuah *project*.

DAFTAR PUSTAKA

<https://www.malasngoding.com/belajar-php-dasar-pengenalan-dan-kegunaan-php/>

<https://idcloudhost.com/panduan/mengenal-sejarah-dan-pengertian-pemrograman-php/>

<https://www.malasngoding.com/php-oop-part-2-pengertian-class-object-property-dan-method/>

<https://www.lautankode.com/apa-itu-constructor-dan-destructor-pada-oop-php/>

<https://jagongoding.com/web/php/apa-itu-composer/>

<https://www.ansoriweb.com/2020/03/pengertian-dfd.html>

<https://glints.com/id/lowongan/dfd-adalah/#.Ybm0gr1By3A>

[https://id.scribd.com/document/428756078/Antarmukapemakai#:~:text=Antarmuka%20pemakai%20\(User%20Interface\)%20merupakan%20mekanisme%20komunikasi%20antara%20pengguna&text=mengarahkan%20alur%20penelusuran%20masalah%20sampai%20ditemukan%20suatu%20solusi.&text=dalam%20mengembangkan%2F%20membangun%20suatu%20program%20jangan%20terlalu%20diperlihatkan.&text=sistem%20operasi%20yang%20bersentuhan%20langsung%20dengan%20pengguna.](https://id.scribd.com/document/428756078/Antarmukapemakai#:~:text=Antarmuka%20pemakai%20(User%20Interface)%20merupakan%20mekanisme%20komunikasi%20antara%20pengguna&text=mengarahkan%20alur%20penelusuran%20masalah%20sampai%20ditemukan%20suatu%20solusi.&text=dalam%20mengembangkan%2F%20membangun%20suatu%20program%20jangan%20terlalu%20diperlihatkan.&text=sistem%20operasi%20yang%20bersentuhan%20langsung%20dengan%20pengguna.)

