BKPM

(BUKU KERJA PRAKTEK MAHASISWA)

PEMROGRAMAN WEB FRAMEWORK
(SEMESTER 4)

OLEH :

TAUFIQ RIZALDI

HERMAWAN ARIEF P.

PROGRAM STUDI MANAJEMEN INFORMATIKA

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI JEMBER

TAHUN 2018

Matakuliah : Pemrograman Web Framework
Minggu Ke : 1
Waktu : 2 x 50 menit
Tema : Konsep Dasar PHP OOP

1. Kompetensi Dasar

a. Mahasiswa mampu memahami konsep dasar PHP OOP
b. Mahasiswa mampu mempraktikkan PHP OOP
c. Mahasiswa mampu memberikan contoh penerapan sistem Informasi

2. Dasar Teori

PHP merupakan bahasa pemrograman yang cukup banyak digunakan untuk membuat web dinamis. Seiring perjalanan waktu PHP terus dikembangkan dan PHP sejak versi PHP 5 telah mendukung Object Oriented Programming atau OOP secara penuh.

Di PHP Object di sini didefinisikan dalam sebuah class. Kemudian Properti object didefinisikan menggunakan kata yang tersisipi var. Sedangkan method dari object berbentuk sebuah function.

3. Alat dan Bahan

Laptop yang sudah terinstal:

a. **Xampp**
b. **Editor notepad ++**

4. Kegiatan Praktikum

4.1 Class

Class merupakan struktur dasar atau sebuah kerangka yang digunakan untuk membentuk sebuah object. Sedangakan Object adalah instance dari class-nya, dengan demikian object itu bisa dikatakan data yang telah terstruktur sesuai dengan yang didefinisikan dalam sebuah class. Jadi di PHP jika Anda ingin membuat object Anda harus mendefinisikan kata class kemudian nama class-nya dibuka dan ditutup menggunakan kurung kurawal {}.

```
1 class Car {
2   // The code
3 }
```

Gambar 1. Contoh Class

4.2 Property

Property dalam suatu class atau object didefinisikan dengan variable. Penulisan variable di dalam class sama seperti penulisan di variable biasa dengan tanda dolar ($), hanya saja variable dalam class Anda harus menyisipkan kata var sebelum dari variable-nya, jika tidak maka akan error di PHP Anda.

```
class Person{var $first_name; // ini adalah sebuah Property }
```

Untuk mengimplementasikan Property Anda cukup menulis object dikuti tanda panah -> kemudian langsung nama property-nya tanpa tanda dolar.

```
1  class Car {
2    public $comp;
3    public $color = 'beige';
4    public $hasSunRoof = true;
5  }
```

Gambar 2. Contoh Property

4.3 Objects

Dan object diimplementasikan dengan variabel dan dikuti kata *new* kemudian baru nama class-nya. Aturan penulisan class adalah Anda boleh menuliskan dengan semua karakter alfabet (abcde s/d z) baik kapital ataupun bukan kapital dan juga karakter underscore (_). Class tidak boleh ditulis dengan numerik (penomoran) kecuali disisipkan diawal dengan karakter alfabet dan class juga tidak boleh ditulis dengan karakter spesial seperti (!?&*;/-{}<> dan lain sebagainya).

```
1  $bmw = new Car ();
2  $mercedes = new Car ();
```

Gambar 3. Contoh Objects

a. Get Object's Properties

```
1  echo $bmw -> color;
2  echo $mercedes -> color;
```

Hasil :

beige

beige

b. Set object's properties

set the color to 'blue' in the bmw object:

```
1 $bmw -> color = 'blue';
```

set value $comp property untuk kedua objects:

```
1 $bmw -> comp = "BMW";
2 $mercedes -> comp = "Mercedes Benz";
```

get the color dari $bmw object

```
1 echo $bmw -> color;
```

Hasil :

Blue

Contoh:

```
1 echo $mercedes -> color;
2 echo $mercedes -> comp;
```

Hasil:

beige

Mercedes Benz

4.4 Methods

Method adalah sebuah funcition yang ditaruh pada class. Method ini merupakan perilaku atau tindakan yang bisa dilakukan terhadap class. Jika property merupakan unsur dari elemen sebuah object maka method lebih kepada bagaimana apa yang bisa dilakukan obyek apa yang bisa dilakukan terhadap obyek seperti memerintahkan dan mengambil data dari obyek.

Untuk membuat method yaitu dengan keyword function diikuti dengan nama method dengan sepasang kurung () untuk menempatkan variable kemudian isi dari method ditaruh diantara dua kurung kurawal {}.

```
class Nama_Class {
        function nama_method () {
         ....// isi method
} }
```

Dengan cara diatas, maka secara otomatis membuat akses dengan public. Atau sama saja dengan cara dibawah ini

```
class Nama_Class {
        public function nama_method () {
         ....// isi method
} }
```

```php
class Car {

  public $comp;
  public $color = 'beige';
  public $hasSunRoof = true;

  public function hello()
  {
    return "beep";
  }
}
```

Gambar 4. Contoh function inside a class = **method**.

Menjalankan method artinya memanggil function dari dalam class. Untuk memanggil method diawali dengan nama object lalu tanda "->" kemudian nama method. Sebuah method bisa menjadi set dan get. Tanda kurung () yang menyertai nama method merupakan tempat untuk menempatkan parameter argument.

```php
$bmw = new Car ();
$mercedes = new Car ();

echo $bmw -> hello();
echo $mercedes -> hello();
```

Gambar 5. Contoh Menjalankan Method

Hasil:

beep

beep

4.8 Contoh

```php
<?php
// Declare class
class Car {
  // properties
  public $comp;
  public $color = 'beige';
  public $hasSunRoof = true;

   // method = hello
  public function hello()
  {
    return "beep";
  }
}

//Membuat instance
$bmw = new Car ();
$mercedes = new Car ();

// Get values
echo $bmw -> color; // beige
echo "<br />";
echo $mercedes -> color; // beige
echo "<hr />";

// Set values
$bmw -> color = 'blue';
$bmw -> comp = "BMW";
$mercedes -> comp = "Mercedes Benz";

// Get values
echo $bmw -> color; // blue
echo "<br />";
echo $mercedes -> color; // beige
echo "<br />";
echo $bmw -> comp; // BMW
echo "<br />";
echo $mercedes -> comp; // Mercedes Benz
echo "<hr />";

// methods get a beep
echo $bmw -> hello(); // beep
echo "<br />";
echo $mercedes -> hello(); // beep
?>
```

5. Latihan

1. Instalasi xampp dan editor notepad ++ (10)
2. Buatlah sebuah kelas bernama ‘buku’, kemudian deklarasikan beberapa properties dari buku tersebut, misalnya: judul buku, pengarang, penerbit,tahun terbit, cetakan. (20)
3. Buatlah kelas kendaraan dengan properties: jenis kendaraan, jumlah roda, merk, bahan bakar, merk, harga, dan tahun pembuatan. Tambahkan fungsi untuk menentukan apakah suatu kendaraan mendapat subsidi BBM atau tidak. Kendaraan yang mendapat subsidi adalah yang berbahan bakar ‘Premium’ dan tahun pembuatannya sebelum tahun

2005. Function ini mereturn ‘Ya’ jika mendapat subsidi, dan ‘Tidak’ jika tidak mendapat subsidi. (20)

4. Buatlah function dalam kelas ‘kendaraan’ dengan nama ‘hargaSecond()’ untuk menentukan harga second dari kendaraan tersebut. Function ini mereturn harga second dari kendaraan dengan ketentuan: (25)
   a. Jika tahun pembuatan di atas 2010, maka harga second nya turun 20% dari harga aslinya
   b. Jika tahun pembuatan 2005 s/d 2010, maka harga second nya turun 30% dari harga aslinya
   c. Jika tahun pembuatan di bawah 2005, maka harga second nya turun 40% dari harga aslinya.
5. Buatlah kasus dimana terdapat class, property (minimal 5), dan function (minimal 3)(25)

Matakuliah : Pemrograman Web Framework

Minggu Ke : 2

Waktu : 2 x 50 menit

Tema : Konsep Dasar PHP OOP (Lanjutan)

1. Kompetensi Dasar

d. Mahasiswa mampu memahami konsep dasar PHP OOP

e. Mahasiswa mampu mempraktikkan PHP OOP

f. Mahasiswa mampu memberikan contoh penerapan sistem Informasi

2. Dasar Teori

Variabel ***$this*** adalah sebuah *variabel khusus* dalam OOP PHP yang digunakan sebagai *penunjuk kepada objek, ketika kita mengaksesnya dari dalam class*. Dalam manual PHP, ***$this*** disebut dengan istilah: ***pseudo-variable***. Kata kunci ***$this*** ini menunjukkan bahwa kita menggunakan methods dan properties milik kelas tersebut, dan memungkinkan kita untuk memiliki akses kepada mereka dalam lingkup kelas ini. Kata kunci ***$this*** memungkinkan kita untuk mendekati properti kelas dan metode dari dalam kelas menggunakan sintaks berikut:

```
1 $this -> propertyName;
2 $this -> methodName();
```

Contoh :

```
1 $this -> comp
```

→ memanggil / mengunakan property comp.

```
1 $this -> color
```

→ memanggil / mengunakan property color.

3. Alat dan Bahan

Laptop yang sudah terinstal:

b. **Xampp**

c. **Editor notepad ++**

4. Kegiatan Praktikum

1. Buat file baru menggunakan notepad++, lalu ketikkan listing program berikut:

```php
<?
class Car {

    //properties
    public $comp;
    public $color = 'beige';
    public $hasSunRoof = true;

    //method = hello
    public function hello()
    {
      return "Beep I am a <i>" . $this -> comp .
        "</i>, and I am <i>" . $this -> color;
    }
}

//create object di class.
$bmw = new Car();
$mercedes = new Car();

// Set values dari class properties.
$bmw -> color = 'blue';
$bmw -> comp = "BMW";
$mercedes -> comp = "Mercedes Benz";

// Call hello method untuk $bmw object.
echo $bmw -> hello();
?>
```

2. Amati dan tuliskan hasil keluaran dari listing program tersebut.

3. Buat file baru dan ketikkan listing program berikut ini, kemudian amati dan tuliskan hasil dari listing program tersebut:

```php
<?php
class Car {

  public $tank;
  // Add gallons of fuel to the tank when we fill it.
  public function fill($float)
  {
    $this-> tank += $float;
    return $this;
  }

  // Substract gallons of fuel from the tank as we ride the car.
  public function ride($float)
  {
    $miles = $float;
    $gallons = $miles/50;
    $this-> tank -= ($gallons);

    return $this;
  }
}

// Create a new object from the Car class.
$bmw = new Car();

// Add 10 gallons of fuel, then ride 40 miles,
// and get the number of gallons in the tank.
$tank = $bmw -> fill(10) -> ride(40) -> tank;

// Printout.
echo "The number of gallons left in the tank: " . $tank . " gal.";
?>
```

5. Latihan.

1. Buat sebuah class bernama class laptop.
2. Pada class laptop tersebut memiliki 2 properties dengan hak akses public, yaitu pemilik dan merk. Pada class tersebut juga memiliki 3 methods, yaitu hidupkan_laptop(), matikan_laptop() dan restart_laptop().
4. Untuk method hidupkan_laptop() berisi teks sebagai berikut:
   "Hidupkan laptop [merk] punya [pemilik]"
5. Untuk method matikan_laptop() berisi teks sebagai berikut:
   "Matikan laptop [merk] punya [pemilik]"
6. Untuk method restart_laptop() berisi teks sebagai berikut:
   "Matikan laptop [merk] punya [pemilik]"
   "Hidupkan laptop [merk] punya [pemilik]"
7. Buatlah 3 objek menggunakan class laptop sehingga menampilkan teks sebagai berikut:
   a. Hidupkan laptop ASUS milik Taufiq.
   b. Matikan laptop Acer milik Arief.
   c. Matikan laptop Lenovo milik Maya. Hidupkan laptop Lenovo milik Maya.

Matakuliah : Pemrograman Web Framework

Minggu Ke : 3

Waktu : 2 x 50 menit

Tema **:** Enkapsulasi dalam PHP OOP

1. Kompetensi Dasar.

a. Mahasiswa mampu memahami konsep dasar PHP OOP
b. Mahasiswa mampu mempraktikkan PHP OOP
c. Mahasiswa mampu memberikan contoh penerapan enkapsulasi dalam PHP OOP

2. Dasar Teori.

Enkapsulasi (***encapsulation***) adalah *sebuah metoda untuk mengatur struktur class dengan cara menyembunyikan alur kerja dari class tersebut*. ***Struktur class*** yang dimaksud adalah **property** dan **method**. Dengan *enkapsulasi*, kita bisa membuat pembatasan akses kepada *property* dan *method*, sehingga hanya *property* dan *method* tertentu saja yang bisa diakses dari luar class. *Enkapsulasi* juga dikenal dengan istilah '***information hiding'***. Dengan *enkapsulasi*, kita bisa memilih *property* dan *method* apa saja yang boleh diakses, dan mana yang tidak boleh diakses. Dengan menghalangi kode program lain untuk mengubah *property* tertentu, *class* menjadi lebih terintegrasi, dan menghindari kesalahan ketika seseorang '*mencoba*' mengubahnya. Programmer yang merancang *class* bisa menyediakan *property* dan *method* khusus yang memang ditujukan untuk diakses dari luar.

3. Alat dan Bahan.

Laptop yang sudah terinstal:

d. **Xampp**
e. **Editor notepad ++**

4. Kegiatan Praktikum.

a. Hak akses public

Ketika sebuah property atau method dinyatakan sebagai public, maka seluruh kode program di luar class bisa mengaksesnya, termasuk class turunan. Berikut ini adalah contoh listing program yang menggunakan hak akses public dalam salah satu propertinya.

```php
<?php

class Car {

  // public methods & properties.
  public $model;

  public function getModel()
  {
    return "The car model is " . $this -> model;
  }
}

$mercedes = new Car();
//akses property dari dalam class
$mercedes -> model = "Mercedes benz";
//akses property dari luar class
echo $mercedes -> getModel();

?>
```

Coba tuliskan listing program tersebut dalam text editor anda, lalu amati dan tuliskan hasilnya.

b. Hak akses private

Hak akses terakhir dalam konsep *enkapsulasi* adalah **private**. Jika sebuah *property* atau *method* di-set sebagai **private**, maka satu-satunya yang bisa mengakses adalah *class* itu sendiri. ***Class*** lain tidak bisa mengaksesnya, termasuk *class turunan*. Akses level **private** sering digunakan untuk menyembunyikan *property* dan *method* agar tidak bisa diakses di luar class. Berikut ini adalah contoh listing program menggunakan hak akses private.

```php
<?php

class Car {

  //private
  private $model;

  public function getModel()
  {
    return "The car model is " . $this -> model;
  }
}

$mercedes = new Car();

////akses property dari luar class.
$mercedes -> model = "Mercedes benz";
echo $mercedes -> getModel();

?>
```

Coba tuliskan listing program tersebut dalam text editor anda, lalu amati dan tuliskan hasilnya.

5. Latihan.

1. Buatlah class kalkulator sederhana menggunakan ketentuan sebagai berikut:

2. Memiliki 3 properti yang digunakan untuk menampung angka yang akan dioperasikan, ketiga property ini memiliki hak akses private.

3. Memiliki 4 methods, yaitu tambah(), kurang(), bagi() dan kali().

4. Buatkan objek yang menggunakan masing-masing methods yang ada dalam class kalkulator.

Matakuliah **:** Pemrograman Web Framework

Minggu Ke **:** 4

Waktu **:** 2 x 50 menit

Tema **:** Inheritance atau Pewarisan/Penurunan

1. **Kompetensi Dasar**
   a. Mahasiswa mampu memahami Inheritance atau Pewarisan/Penurunan
   b. Mahasiswa mampu mempraktrikkan Inheritance atau Pewarisan/Penurunan
   c. Mahasiswa mampu memberikan contoh pada penerapan sistem Informasi
2. **Dasar Teori**

   Inheritance atau Pewarisan/Penurunan adalah konsep pemrograman dimana sebuah class dapat ‘menurunkan’ property dan method yang dimilikinya kepada class lain. Konsep inheritance digunakan untuk memanfaatkan fitur ‘code reuse’ untuk menghindari duplikasi kode program Konsep inheritance membuat sebuah struktur atau ‘hierarchy’ class dalam kode program. Class yang akan ‘diturunkan’ bisa disebut sebagai class induk (parent class), super class, atau base class. Sedangkan class yang ‘menerima penurunan’ bisa disebut sebagai class anak (child class), sub class, derived class atauheir class. Tidak semua property dan method dari class induk akan diturunkan. Property dan method dengan hak akses private, tidak akan diturunkan kepada class anak. Hanya property dan method dengan hak akses protected dan public saja yang bisa diakses dari class anak..
3. **AlatdanBahan**

   Laptop yang sudahterinstal:

   a. Xampp

   b. Editor notepad ++
4. **KegiatanPraktikum**

4.1 Inheritance

Inheritance dalam PHP berorientasi objek mengurangi duplikasi kode, Inheritance memungkinkan kita untuk menulis Code hanya sekali pada induknya (parent class), Super class atau base class dan kemudian menurunkan atau digunakan pada class anak (child class), sub class, atau derived class.

```php
<?php
//The parent class
class Car {
  // Private property inside the class
  private $model;

  //Public setter method
  public function setModel($model)
  {
    $this -> model = $model;
  }

  public function hello()
  {
    return "beep! I am a <i>" . $this -> model . "</i><br />";
  }
}

//The child class inherits the code from the parent class
class SportsCar extends Car {
  //No code in the child class
}

//Create an instance from the child class
$sportsCar1 = new SportsCar();

// Set the value of the class' property.
// For this aim, we use a method that we created in the parent
$sportsCar1 -> setModel('Mercedes Benz');

//Use another method that the child class inherited from the parent class
echo $sportsCar1 -> hello();
?>
```

Gambar 6. Contoh Inheritance

4.2 Enkapsulasi Objek : Private dan Protected

Enkapsulasi (encapsulation) adalah sebuah metoda untuk mengatur struktur class dengan cara menyembunyikan alur kerja dari class tersebut. Struktur class yang dimaksud adalah property dan method. Dengan enkapsulasi, kita bisa membuat pembatasan akses kepada property dan method, sehingga hanya property dan method tertentu saja yang bisa diakses dari luar class. Enkapsulasi juga dikenal dengan istilah 'information hiding'

konsep enkapsulasi : **private** memungkinkan Jika sebuah property atau method di-set sebagai private, maka satu-satunya yang bisa mengakses adalah class itu sendiri. Class lain tidak bisa mengaksesnya, termasuk class turunan.

```php
<?php
// The parent class
class Car {
    //The $model property is private, thus it can be accessed
    // only from inside the class
    private $model;
    //Public setter method
    public function setModel($model)
    {
        $this -> model = $model;
    }
}
// The child class
class SportsCar extends Car{
    //Tries to get a private property that belongs to the parent
    public function hello()
    {
        return "beep! I am a <i>" . $this -> model . "</i><br />";
    }
}
//Create an instance from the child class
$sportsCar1 = new SportsCar();
//Set the class model name
$sportsCar1 -> setModel('Mercedes Benz');
```

Gambar 7. Contoh Private

Jika sebuah *property* atau *method* dinyatakan sebagai **protected**, berarti *property* atau *method* tersebut *tidak bisa diakses dari luar class, namun bisa diakses oleh class itu sendiri atau turunan class tersebut*.

```php
<?php
// The parent class
class Car {
    //The $model property is now protected, so it can be accessed
    // from within the class and its child classes
    protected $model;

    //Public setter method
    public function setModel($model)
    {
        $this -> model = $model;
    }
}

// The child class
class SportsCar extends Car {
    //Has no problem to get a protected property that belongs to the parent
    public function hello()
    {
        return "beep! I am a <i>" . $this -> model . "</i><br />";
    }
}

//Create an instance from the child class
$sportsCar1 = new SportsCar();
```

Gambar 3. Contoh Protected

4.3 Abstract

Abstract Class adalah sebuah class yang tidak bisa di-instansiasi (tidak bisa dibuat menjadi objek) dan berperan sebagai ‘kerangka dasar’ bagi class turunannya. Di dalam abstract class umumnya akan memiliki abstract method.

```php
abstract class Car {
  // Abstract classes can have properties
  protected $tankVolume;

  // Abstract classes can have non abstract methods
  public function setTankVolume($volume)
  {
    $this -> tankVolume = $volume;
  }

  // Abstract method
  abstract public function calcNumMilesOnFullTank();
}
```

Gambar 4. Contoh Absrract

4.4 Overriding

Overriding Merupakan Suatu Keadaan Dimana kelas anak dapat mengubah atau bisa kita bilang memodifikasi atau memperluas data dan method pada kelas induk. Keuntungan Overriding : dapat menambahkan sifat / atribut pada kelas induk nya.

```php

//Set the class model name
$sportsCar1 -> setModel('Mercedes Benz');

//Get the class model name
echo $sportsCar1 -> hello();
?>
```

Gambar 5. Contoh Class Induk

Selanjutnya melakukan override property dan method yang dimiliki induk class pada child class.

```php
// The parent class has hello method that returns "beep".
class Car {
  public function hello()
  {
    return "beep";
  }
}

//The child class has hello method that returns "Halllo"
class SportsCar extends Car {
  public function hello()
  {
    return "Hallo";
  }
}

//Create a new object
$sportsCar1 = new SportsCar();

//Get the result of the hello method
echo $sportsCar1 -> hello();
```

Gambar 8. Contoh Override

Selanjutnya mencegah override pada child class dari parent class

```php
// The parent class has hello method that returns "beep".

class Car {
  final public function hello()
  {
    return "beep";
  }
}

//The child class has hello method that tries to override the hello method in the parent
class SportsCar extends Car {
  public function hello()
  {
    return "Hallo";
  }
}


//Create a new object
$sportsCar1 = new SportsCar();

//Get the result of the hello method
echo $sportsCar1 -> hello();
```

Gambar 6. Contoh Prevent Override

4.5 Abstract Classes and Methods

Kita menggunakan class abstract dan methods ketika kita perlu melakukan child class dengan methods tertentu yang mewarisi dari class parent. class abstrak adalah class yang memiliki setidaknya satu methods abstrak. methods abstrak hanya dapat memiliki name dan arguments, dan tidak ada code lain. Dengan demikian, kita tidak bisa membuat objek dari class abstrak. Sebaliknya, kita perlu membuat child class yang menambahkan code ke dalam tubuh methods, dan menggunakan child class ini untuk membuat objek.

Deklarasi classes dan methods abstract

```php
// Abstract classes are declared with the abstract keyword, and contain abstract methods.
abstract class Car {
  abstract public function calcNumMilesOnFullTank();
}
```

Gambar 7. Contoh class abstract

Tidak satupun methods didalam class abstract

5. Latihan

1. Instalasi xampp dan editor notepad ++ (10)
2. Buatlah sebuah kelas bernama ‘MobilBMW’, yang merupakan inherit dari class ‘mobiLengkap’ kemudian deklarasikan beberapa methods dari ‘mobilLengkap’ tersebut, misalnya: ‘nontonTV’ yang isinya menampilkan Tv dihidupkan, Tv Mencari Chanel, Tv Menampilkan gambar. Kemudian buat class ‘MobilBMWberaksi’ yang didalamnya terdapat methods ‘nontonTv’, ‘HidupkanMobil’, ‘MatikanMobil’, ‘ubahGigi’
3. Buatlah class Topi, class Celana, Class baju yang memiliki property dan methods berdasakan fakta misalkan topi dengan $model , celana $tipe $model dan baju $tipe kemudian tiga class tersebut extends class item produk yang memiliki methods Ukuran, Warna, Nama .
4. Buatlah class Tablet dengan beberapa property dan sebuah method didalamnya. Property class tablet $merk, $camera, dan $memory Kemudian buat class handphone mewarisi class tablet. dalam class handphone bisa mengakses seluruh property dan method apapun dari class tablet misalkan ditambahkan $handphone_baru dari class handphone dan Method beli_handphone(). Buatlah tiga code program yang berbeda tetapi menjalakan enkapsulasi model pada objek : Public , Protect, Private yang mewariskan kelass induk (kemudian jelaskan Perbedaanya).

Matakuliah : Pemrograman Web Framework

Minggu Ke : 5

Waktu : 2 x 50 menit

Tema : Interface dan Polymorfisme dalam PHP OOP

1. Kompetensi Dasar

a. Mahasiswa mampu memahami konsep dasar PHP OOP

b. Mahasiswa mampu mempraktikkan PHP OOP

c. Mahasiswa mampu memberikan contoh penerapan penggunaan interface dan polymorfisme dalam pemrograman OOP.

2. Dasar Teori

2.1 Interface

Secara sederhana, Object Interface adalah sebuah ‘kontrak’ atau perjanjian implementasi method.Bagi class yang menggunakan object interface, class tersebut harus mengimplementasikan ulang seluruh method yang ada di dalam interface. Dalam pemrograman objek, penyebutan object interface sering disingkan dengan ‘Interface’ saja. Jika anda telah mempelajari abstract class, maka interface bisa dikatakan sebagai bentuk lain dari abstract class. Walaupun secara konsep teoritis dan tujuan penggunaannya berbeda. Sama seperti abstract class, interface juga hanya berisi signature dari method, yakni hanya nama method dan parameternya saja (jika ada). Isi dari method akan dibuat ulang di dalam class yang menggunakan interface. Jika kita menganggap abstract class sebagai ‘kerangka’ atau ‘blue print’ dari class-class lain, maka interface adalah implementasi method yang harus ‘tersedia’ dalam sebuah objek. Interface tidak bisa disebut sebagai ‘kerangka’ class.

a. Bentuk umum dari penulisan interface

```
interface interfaceName {
  // abstract methods
}

class Child implements interfaceName {
  // defines the interface methods and may have its own code
}
```

Contoh penerapan:

```php
interface Car {
  public function setModel($name);

  public function getModel();
}
```

b. Perbedaan *Interface* dengan *abstract class.*

	interface	abstract class
the code	- abstract methods - constants	- abstract methods - constants - concrete methods - concrete variables
access modifiers	- public	- public - protected - private etc.
number of parents	The same class can implement more than 1 interface	The child class can inherit only from 1 abstract class

2.2 Polymorfisme

Dari segi bahasa, Polimorfisme (bahasa inggris: Polymorphism) berasal dari dua kata bahasa latin yakni poly dan morph. Poly berarti banyak, dan morph berarti bentuk. Polimorfisme berarti banyak bentuk (wikipedia). Di dalam pemrograman objek, polimorfisme adalah konsep dimana terdapat banyak class yang memiliki signature method yang sama. Implementasi dari method-method tersebut diserahkan kepada tiap class, akan tetapi cara pemanggilan method harus sama. Agar kita dapat 'memaksakan' signature method yang sama pada banyak class, class tersebut harus diturunkan dari sebuah abstract class atau object interface.

3. Alat dan Bahan

Laptop yang sudah terinstal:

a. Xampp

b. Editor notepad ++

4. Kegiatan Praktikum

1. Buatlah file baru pada text editor masing-masing, dan tuliskan listing program berikut ini:

```php
interface Car {
  public function setModel($name);

  public function getModel();
}
```

```php
class miniCar implements Car {
  private $model;

  public function setModel($name)
  {
    $this -> model = $name;
  }

  public function getModel()
  {
    return $this -> model;
  }
}
```

2. Buatlah 3 object baru dari class minicar, kemudian masukkan nilai ‘multi-purpose vehicle’, ‘sedan’ dan ‘hatchback’ sebagai nama modelnya. Tuliskan dan amati hasilnya.
3. Buat file baru pada text editor masing-masing kemudian tuliskan listing program berikut:

```php
interface Shape {
  public function calcArea();
}
```

```php
class Circle implements Shape {
  private $radius;

  public function __construct($radius)
  {
    $this -> radius = $radius;
  }

  // calcArea calculates the area of circles
  public function calcArea()
  {
    return $this -> radius * $this -> radius * pi();
  }
}
```

```php
<?
class Rectangle implements Shape {
  private $width;
  private $height;

  public function __construct($width, $height)
  {
    $this -> width = $width;
    $this -> height = $height;
  }

  // calcArea calculates the area of rectangles
  public function calcArea()
  {
    return $this -> width * $this -> height;
  }
}

$circ = new Circle(3);
$rect = new Rectangle(3,4);

echo $circ -> calcArea();
echo $rect -> calcArea();
?>
```

5. Latihan

1. Buatlah file text baru pada text editor masing-masing.
2. Buatlah interface yang bernama hitungLuas dengan property bernama ‘sisi’ dan berisi 3 methods yaitu fungsi hitungLuasPersegi(), hitungLuasSegitiga() dan hitungLuasLingkaran().
3. Buatlah 3 class untuk menghitung luasbangun datar yang mengimplementasikan interface tersebut.
4. Buatlah object dari masing-masing class, kemudian jalankan dan amati hasilnya.

Matakuliah **:** Pemrograman Web Framework
Minggu Ke **:** 6
Waktu **:** 2 x 50 menit
Tema **:** Web Service

1. Kompetensi Dasar

a. Mahasiswa mampu memahami konsep dasar PHP OOP
b. Mahasiswa mampu mempraktikkan PHP OOP
c. Mahasiswa mampu memberikan contoh penerapan web service menggunakan PHP

2. Dasar Teori

Web service adalah aplikasi sekumpulan data (*database*), perangkat lunak (*software*) atau bagian dari perangkat lunak yang dapat diakses secara remote oleh berbagai piranti dengan sebuah perantara tertentu. Secara umum,*web service* dapat diidentifikasikan dengan menggunakan URL seperti hanya web pada umumnya. Namun yang membedakan *web service* dengan web pada umumnya adalah interaksi yang diberikan oleh *web service*. Berbeda dengan URL *web* pada umumnya, URL *web service* hanya menggandung kumpulan informasi, perintah, konfigurasi atau sintaks yang berguna membangun sebuah fungsi-fungsi tertentu dari aplikasi. *Web service* dapat diartikan juga sebuah metode pertukaran data, tanpa memperhatikan dimana sebuah*database* ditanamkan, dibuat dalam bahasa apa sebuah aplikasi yang mengkonsumsi data, dan di platform apa sebuah data itu dikonsumsi. *Web service* mampu menunjang interoperabilitas. Sehingga *web service* mampu menjadi sebuah jembatan penghubung antara berbagai sistem yang ada. Menurut W3C *Web services Architecture Working Group* pengertian *Web service* adalah sebuah sistem *software*yang di desain untuk mendukung interoperabilitas interaksi mesin ke mesin melalui sebuah jaringan. Interface*web service* dideskripsikan dengan menggunakan format yang mampu diproses oleh mesin (khususnya WSDL). Sistem lain yang akan berinteraksi dengan *web service* hanya memerlukan SOAP, yang biasanya disampaikan dengan HTTP

dan XML sehingga mempunyai korelasi dengan standar Web (Web Services Architecture Working Group, 2004).

Web pada umumnya digunakan untuk melakukan *respon* dan *request* yang dilakukan antara *client* dan *server*. Sebagai contoh, seorang pengguna layanan *web* tertentu mengetikan alamat *url web* untuk membentuk sebuah*request*. *Request* akan sampai pada *server*, diolah dan kemudian disajikan dalam bentuk sebuah respon. Dengan singkat kata terjadilah hubungan *client-server* secara sederhana. Sedangkan pada *web service* hubungan antara *client* dan *server* tidak terjadi secara langsung. Hubungan antara*client* dan *server* dijembatani oleh file *web service* dalam format tertentu. Sehingga akses terhadap *database*akan ditanggani tidak secara langsung oleh *server*, melainkan melalui perantara yang disebut sebagai *web service*. Peran dari *web service* ini akan mempermudah distribusi sekaligus integrasi database yang tersebar di beberapa *server* sekaligus.

3. Alat dan Bahan

Laptop yang sudah terinstal:

a. Xampp

b. Editor notepad ++

4. Kegiatan Praktikum

1. Sebelumnya buatlah database sederhana yang bernama db_bukuTamu yang berisi 3 field yaitu; Id_buku_tamu, int 5; nama_tamu varchar 15; alamat_email varchar 25. Isikan record berupa 5 nama anda dan teman anda kedalam database
2. Tuliskan listing program berikut ini kemudian simpan dengan nama koneksi.php (sesuaikan username, password dan nama database dengan milik kalian).

```php
<?php
 $host = "localhost";  //Nama Host
 $user = "root"; //Nama User
 $pass = ""; //Password
 $db = "............."; //Nama Database

 //Koneksi
 mysql_connect($host, $user, $pass)
  or die (mysql_error());

 //Pilih Database
 mysql_select_db($db)
  or die(mysql_error()." Database Not Found!");
?>
```

3. Kemudian tuliskan listing program berikut dan simpan dengan nama webservis.php, amati dan analisa hasilnya.

```php
<?php
 //Memanggil file koneksi
 include "file koneksi.php";

 //Syntax MySql untuk melihat semua record yang
 //ada di tabel
 $sql = "SELECT * FROM <nama_tabel>";

 //Execetute Query diatas
 $query = mysql_query($sql);
 while($dt=mysql_fetch_array($query)){
  $item[] = array(
   "kolom_1"=>$dt["kolom_1"],
   "kolom_2"=>$dt["kolom_2"],
   "kolom_3"=>$dt["kolom_3"],
   ..... dst
    );
 }

 //Menampung data yang dihasilkan
 $json = array(
    'result' => 'Success',
    'item' => $item
   );

 //Merubah data kedalam bentuk JSON
 echo json_encode($json);
?>
```

5. Latihan

1. Buatlah database tentang peminjaman buku yang berisi minimal tiga tabel, yaitu tabel buku, tabel anggota dan tabel peminjaman, yang mana untuk tabel peminjaman berupa tabel transaksi untuk tabel anggota dan tabel buku. Kemudian isikan minimal 6 record kedalam tiap tabel.
2. Buatlah file koneksi dan web servis untuk menampilkan data dari masing-masing tabel dalam database.

3. Buatlah user interface sederhana untuk mengisikan data kedalam database melalui web.

Matakuliah : Pemrograman Web Framework

Minggu Ke : 7

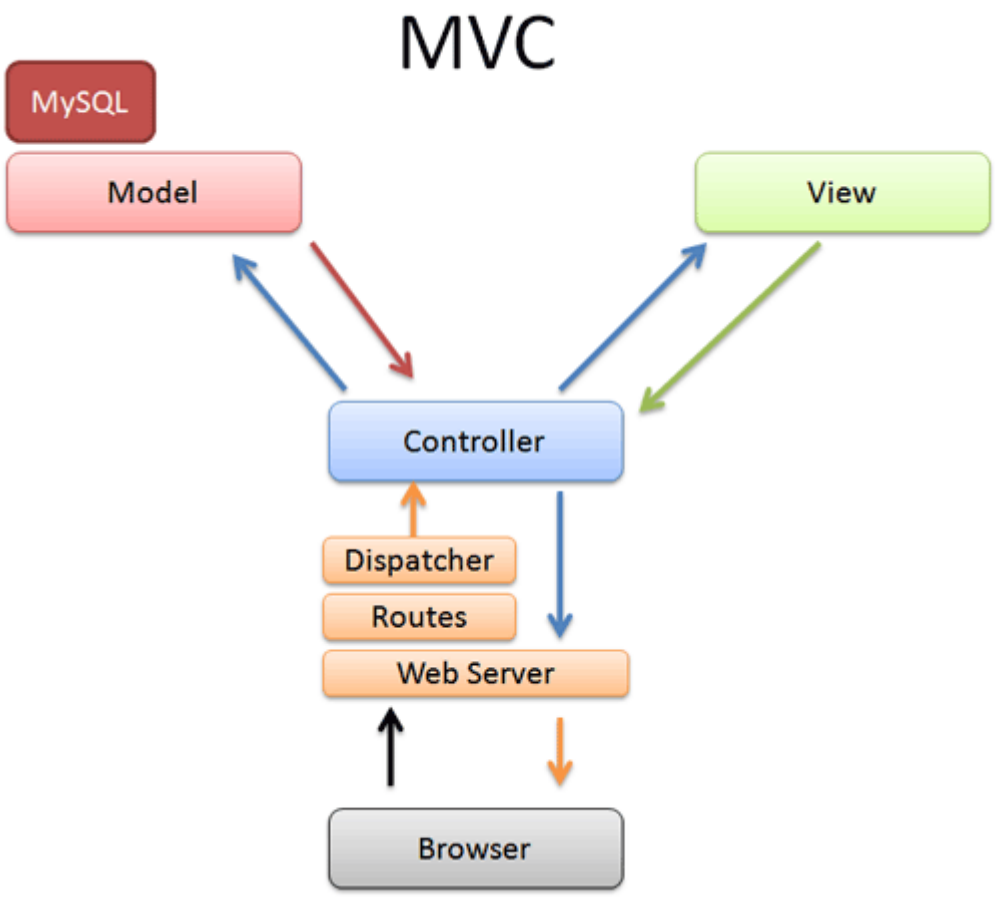
Waktu : 2 x 50 menit

Tema : MVC

1. **Kompetensi Dasar**
   a. Mahasiswa mampu memahami konsep dasar MVC
   b. Mahasiswa mampu memberikan contoh penerapan MVC
2. **Dasar Teori**

   Konsep MVC

   Konsep MVC memisahkan antara proses pengolahan data ke database (Model), Tempat Pengolahan Request atau bussines logic aplikasinya di Controller, dan Penampilan datanya hanya di View. skemanya seperti berikut ini



ketika kita megetikan sebuah alamat semisal http://ci.com, Proses itu langsung di ambil alih oleh controller, setelah itu di controller jika ada data yang ingin di tampilkan dari database, dia akan memanggil model dan menanyakan data, jika datanya ada, maka akan di kembalikan lagi di controller kemudian diolah di controller setelah itu baru data tersebut di kirim ke View dan siap di tampilkan.

3. **Alat dan Bahan**

   Laptop yang sudah terinstal:

   a. **Xampp**

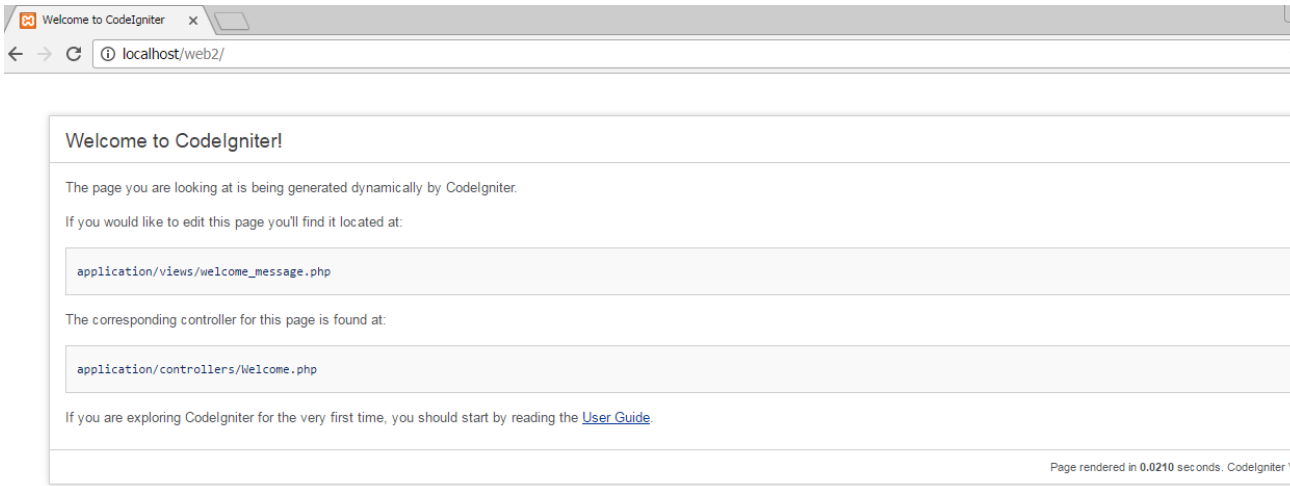
   b. **Editor**

   c. **Code Igniter 3.0.0**

4. **Kegiatan Praktikum**

   Code Igniter

   Jika Instalasi berhasil



Note :untuk mengakses halaman suatu aplikasi yang dibangun dengan code igniter url yg diketik adalah

<nama folder ci>/index.php/<nama controller>

Contoh : localhost/web2/index.php/hello

Cobalah kode – kode dibawah ini

Contoh1 : Controller

```php
<?php
class Hello extends CI_Controller{
    public function index(){
        echo "<h2>Hello World CI!</h2>";
    }
}
?>
```

Contoh2 : Controller dan View

File view : helloview.php

```php
<html>
<head></head>
<body>
    <h2>Hello World CI!</h2>
    <h3>Menggunakan Controller dan View!</h3>
</body>
</html>
```

File Controller : Hello.php

```php
<?php
class Hello extends CI_Controller{
    public function index(){
        //echo "<h2>Hello World CI!</h2>";

        //memanggil file view
        $this->load->view('helloview');//file view
    }
}
?>
```

Contoh3 : Controller, View dan Model

Controller : Hello.php

```php
<?php
class Hello extends CI_Controller{
    public function index(){
        //c3 - memuat model 'Hello_model'
        $this->load->model('Hello_model');

        //Pengambilkan objek dari kelas Hello_model dan dimuat di var $model
        $model = $this->Hello_model;

        //Mengambil data dari model
        $a = $model->txt;

        //Membuat data yang akan dikirim ke view
        $data['teks'] = $a;

        //memanggil file view
        $this->load->view('helloview', $data);//file view
```

Model : Hello_model.php

```php
<?php
    class Hello_model extends CI_Model{
        //membuat properti dengan nama var $txt
        public $txt = 'Hello World!';
    }
?>
```

View : helloview.php

```php
<html>
<!--<head>Controller dan View</head>-->
<head><title>Controller, Model dan View</title></head>
<body>
    <h2><? echo $teks; ?></h2>
    <h3>Menggunakan Controller, Model dan View!</h3>
</body>
</html>
```

Contoh4 ; Controller dan view

Controller

```php
<?php
class Variabel extends CI_Controller{
    public function index(){
        $data = ['variabel1'=>'Data variabel ke 1', 'variabel2'=>'Data variabel ke 2'];

        $this->load->view('variabelview', $data);
    }
}

?>
```

View

```
<html>
<head><title>Controller dan View lebih dari 1 Variavbel</title></head>
<body>
    <h2>Menggirim Data dari Controller ke View</h2>

    <!--Memanggil variabel 1-->
    variabel1: <?php echo variabel1; ?></br>

    <!--Memanggil variabel 2-->
    variabel2: <?php echo variabel2; ?></br>
</body>
</html>
```

5. Latihan

a. Buatlah sebuah halaman yang menampilkan pesan “Hello Wolrd dari CI Model” dimana pesan tersebut terdapat di dalam model dan dipanggil dari controller.

b. Buatlah sebuah tampilan yang mengirim lebih dari 1 variabel menggunakan Controller, Model danView.

Matakuliah : Pemrograman Web Framework

Minggu Ke : 8

Waktu : 2 x 50 menit

Tema : MVC (Lanjutan)

1. **Kompetensi Dasar**
   a. Mahasiswa mampu memahami konsep dasar MVC
   b. Mahasiswa mampu memberikan contoh penerapan MVC
2. **Dasar Teori**

   -
3. **Alat dan Bahan**

   Laptop yang sudah terinstal:

   a. Xampp

   b. Editor

   c. Code Igniter 3.0.0
4. **Kegiatan Praktikum**

   View

   Load multiple view

   headerview.php

```
<html>
<head>
<title>Demo View</title>
</head>
<body>
<h1>Multiple View </h1>
```

contentview.php

```
<p>Contoh penerpan multiple view yang dipanggil dari satu controller</P>
```

footerview.php

```
<hr/>
Copyright : Footer
</body>
</html>
```

Demo_view.php

```php
<?php
class Demo_view extends CI_Controller{
    public function index(){
        $this->load->view('headview');
        $this->load->view('contentview');
        $this->load->view('footerview');
    }
}
?>
```

Menyisipkan CSS ke View 1

Demoview.php

```php
<html>
<head>
<title>Demo View CSS</title>
<style type="text/css">
    h2{
        color: #f00;
        font-size: 20px;
        border-bottom: dashed 1px #f00;
    }
    p{
        font-sytle: italic;
    }
    <!--<link rel="stylesheet" type="text/css" href="<?php //echo base_url(); ?>">-->
</style>
</head>
<body>
    <h2>Demo View dengan CSS</h2>
    <p>contoh demo view dengan css</p>
</body>
</html>
```

Demo_view2.php

```php
<?php
class Demo_view2 extends CI_Controller{
    public function index(){
        $this->load->view('demoview');
    }
}
?>
```

Menyisipkan CSS ke View 2

demoview.php

```
<html>
<head>
<title>Demo View CSS</title>
<!--<style type="text/css">
    h2{
        color: #f00;
        font-size: 20px;
        border-bottom: dashed 1px #f00;
    }
    p{
        font-sytle: italic;
    }
</style>-->
<link rel="stylesheet" type="text/css" href="<?php //echo base_url(); ?>">
</head>
<body>
    <h2>Demo View dengan CSS</h2>
    <p>contoh demo view dengan css</p>
</body>
</html>
```

Demo_view2.php

```
<?php
class Demo_view2 extends CI_Controller{
    public function __construct(){
        parent:: __construct();
        $this->load->helper('url')

    }

    public function index(){
        $this->load->view('demoview');
    }
}
?>
```

style.css→diletakn di folder assests\css *jikatidakadaharusdibuatdulu

```
h2{
        color: #f00;
        font-size: 20px;
        border-bottom: dashed 1px #f00;
    }
    p{
        font-sytle: italic;
    }
```

Controller

Demo_controller.php

```php
<?php

class Demo_controller extends CI_Controller{

    public function index(){
        echo "<h2>Demo Controller</h2>";
        echo "<br>Function yang dipanggil adalah index";
    }
    public function aksi(){
        echo "<h2>Demo Controller</h2>";
        echo "<br>Function yang dipanggil adalah aksi";
    }
}
```

Melewatkan segment URI kedalamMetode

Kode.php

```php
<?php
class Kode extends CI_Controller{

    public function hello($var){
        if(isset($var)){
            switch(strtolower($var)){
            case 'php':
                $this->load->view('phpview');
                break;
            case 'python':
                $this->load->view('pythonview');
                break;
            case 'cpp':
                $this->load->view('cppview');
                break;
            case 'java':
                $this->load->view('javaview');
                break;
            default:
                echo 'Input URI Salah';
            }
        }else{
            echo 'Input URI Salah';
        }
    }
}
```

phpview.php

```
<html>
<head>
 <title>Hello View</title>
</head>
<body>
 <h2>"Hello Wolrd! PHP"</h2>

</body>
</html>
```

cppview.php

```
<html>
<head>
 <title>Hello View</title>
</head>
<body>
 <h2>"Hello Wolrd! CPP"</h2>

</body>
</html>
```

pythonview.php

```
<html>
<head>
 <title>Hello View</title>
</head>
<body>
 <h2>"Hello Wolrd! Python"</h2>

</body>
</html>
```

javaview.php

```
<html>
<head>
 <title>Hello View</title>
</head>
<body>
 <h2>"Hello Wolrd! Java"</h2>

</body>
</html>
```

Memetakan nama metode yang akandipanggil

```php
<?php
class Kode1 extends CI_Controller{

    public function index(){
        $this->load->view('phpview');
    }
    public function hello_python(){
        $this->load->view('pythonview');
    }
    public function hello_cpp(){
        $this->load->view('cppview');
    }
    public function hello_java(){
        $this->load->view('javaview');
    }
    public function _remap($var){
        if(isset($var)){
            switch(strtolower($var)){
            case 'python':
                $this->hello_python();
                break;
            case 'cpp':
                $this->hello_cpp();
                break;
            case 'java':
                $this->hello_java();
                break;
            default:
                $this->index();
            }
        }else{
            $this->index();
        }
    }
}
```

Matakuliah : Pemrograman Web Framework

Minggu Ke : 9

Waktu : 2 x 50 menit

Tema : Model

1. **Kompetensi Dasar**
   a. Mahasiswa memahami Model
   b. Mahasiswa mampu menerapkan penggunan Model
2. **Dasar Teori**

   Model mewakili struktur datadari website yang bisa berupa basis data maupun data lain, misalnya dalam bentuk file teks atau file xml. Biasanya didalam model akan berisi class dan fungsi untuk mengambil, melakukan update dan menghapus data website. Karena sebuah website biasanya menggunakan basis data dalam menyimpan data maka bagian Model biasanya akan berhubungan dengan perintah-perintah query SQL. Model bisa dibilang khusus digunakan untuk melakukan koneksi ke basis data oleh karena itu logika-logika pemrograman yang berada didalam model juga harus yang berhubungan dengan basis data. Misalnya saja pemilihan kondisi tetapi untuk memilih melakukan query yang mana. Bentuk umum Model adalah sebagai berikut:

```
class Persegipanjang_model extends CI_model{
        //badan class
        // Konstruktor kelas model
        Function __construct();
parent::_construct();
}
```

Note : nama model harus sama dengan nama class- nya

Contoh :

```php
<?php
class temp_model extends Model {
    function temp_model()
    {
        parent::Model();
    }

    function getProducts()
    {
        return $this->db->query("SELECT * FROM AA_PRODUCTS");
    }
}
?>
```

3. Alat dan Bahan

Laptop yang sudah terinstal:

a. Xampp

b. CodeIgniter

c. Editor notepad ++

4. Kegiatan Praktikum

4.1. C1:

Persegipanjang_model.php

```php
<?php
class Persegipanjang_model extends CI_model{
    //atribut model
    private $panjang;
    private $lebar;

    //menentukan nilai $panjang dan $lebar
    public function set_panjang($p){
        $this->panjang = $p;
    }
    public function set_lebar($l){
        $this->lebar = $l;
    }

    //pengambilan nilai $panjang dan $lebar
    public function get_panjang(){
        return $this->panjang;
    }
    public function get_lebar(){
        return $this->lebar;
    }

    //perhitungan luas persegi panjang
    public function hitung_luas(){
        return $this->panjang * $this->lebar;
    }
    //perhitungan keliling persegi panjang
    public function hitung_keliling(){
        return 2 * $this->panjang * $this->lebar;
    }
}
```

Persegipanjang.php

```php
<?php
class Persegipanjang extends CI_Controller{
    public function index(){
    //pemanggilan model
    $this->load->model('Persegipanjang_model');
    $model = $this->Persegipanjang_model;

    //penentuan nilai panjang & lebar
    $model->set_panjang(4);
    $model->set_lebar(5);

    //menampilkan ke view
    $this->load->view('Persegipanjangview', array('model'=>$model));
    }
}
```

Persegipanjangview.php

```php
<html>
<head>
<title>Latihan Model</title>
</head>
<body>
    <h1>Model Persegi Panjang</h1>

    <!--Pemanggilan metode get_panjang dan get_lebar-->
    Nilai Panjang: <?php echo $model->get_panjang(); ?></br>
    Nilai lebar: <?php echo $model->get_lebar(); ?></br>

    <!--Pemanggilan metode hitung_luas-->
    Luas Persegi Panjang: <?php echo $model->hitung_luas(); ?></br>
    <!--Pemanggilan metode hitung_keliling-->
    Keliling Persegi Panjang: <?php echo $model->hitung_keliling(); ?></br>
</body>
</html>
```

4.2. Menggunakan Inputan

1.Gunakan model yang sama (Persegipanjang_model.php)

2.Buatlah persegipanjangfromview.php

```php
<html>
<head>
<title>Latihan Model</title>
</head>
<body>
    <h1>Model Persegi Panjang</h1>
        <form action="http://localhost/nama_folder/index.php/persegipanjang" method="post">
            Masukan nilai Panjang : <input type="text" name="p"/><br><br>
            Masukan nilai Lebar   : <input type="text" name="l"/><br>
            <input type="submit" name="btnSubmit" value="Hitung"><br>
        </form>
</body>
</html>
```

3.Rubah controller Persegipanjang.php seperti dibawah ini

```php
class Persegipanjang extends CI_Controller{
    public function index(){
        if(isset($_POST['...nama tombol...'])){
            //pemanggilan / pembuatan model
            ....................................
            ....................................

            //pengambilan nilai inputan
            $panjang = $_POST['p'];
            $lebar = $_POST['l'];

            //set nilai panjang dan lebar dengan fungsi set_.....
            ....................................
            ....................................

            //pemangilan view
            $this->load->view('persegipanjangview', array('model'=>$model));
        }else{
            //menampilkan form
            $this->load->view('persegipanjangformview');
        }
    }
}
```

4.3. Contoh Model untuk form

User_model.php

```php
<?php
class User_model extends CI_Model{
    public $username;
    public $password;

    public $labels = [];

    public function __construct(){
        parent::__construct();
        $this->labels = $this->attribut_labels();
    }

    public function aunthentikasi(){
        if(isset($this->username) && isset($this->password)){
            if ($this->username=== 'demo' && $this->password === 'demo'){
                return TRUE;
            }else{
                return FALSE;
            }
        }else{
            return FALSE;
        }
    }
    public function attribute_labels(){
        return['username'=>'Username:','password'=>'Password:'];
    }
}
```

Login.php

```php
<?php
class Login extends CI_Controller{
    public function index(){
        $this->load->model('User_model');
        $model = $this->User_model;

        if(isset($_POST['btnSubmit'])){
            $model->username = $_POST['username'];
            $model->password = $_POST['password'];

            if($model->aunthentikasi()){
                $this->load->view('login_success_view');
            }else{
                $this->load->view('login_error_view');
            }
        }else{
            $this->load->view('login_form_view', ['model'=>'$model']);
        }
    }
}
```

login_form_view.php

```php
<html>
<head>
 <title>Demo Models</title>
</head>
<body>
 <h2>Login</h2>
<form action="http://localhost/web2/index.php/login" method=POST>
 <?php echo $model->labels['username']; ?><br>
 <input type = 'text' name = 'username'><br><br>
 <?php echo $model->labels['password']; ?><br>
 <input type = 'text' name = 'password'><br><br>
 <input type = 'submit' name = 'btnSubmit' value="Login"><br><br>
</form>
</body>
</html>
```

login_seccess_view.php

```php
<html>
<head>
 <title>Demo Models</title>
</head>
<body>
 <h2>Login Success</h2>
</body>
</html>
```

login_error_view.php

```php
<html>
<head>
 <title>Demo Models</title>
</head>
<body>
 <h2>Login Error</h2>
</body>
</html>
```

4.4. Model dengan database

Konfigurasi db di CI

Application\config\database.php

```php
$db['default'] = array(
    'dsn'   => '',
    'hostname' => 'localhost', //Masukan hostname disini default : "localhost"
    'username' => 'root',      //Masukan usename  disini default : "root"
    'password' => '',          //Masukan usename  disini default : ""
    'database' => 'db_ci',     //Masukan nama db yg akan digunakan contoh : "db_ci"
```

Buatlah tabel contoh : tabel mahasiswa

No	Field	Type
1	nim	Char(8)
2	nama	Varchar(30)
3	prodi	Varchar(30)

Mahasiswa_model.php

```php
<?php
class Mahasiswa_model extends CI_Model{
    public $nim;
    public $nama;
    public $prodi;

    public $labels = [];

    public function __construct(){
        parent::__construct();
        $this->labels = $this->attribute_labels();
        $this->load->database();
    }

    public function get_table_name(){
        return 'product';
    }

    public function insert(){
        $this->db->insert($this->get_table_name(). $this);
    }

    public function update(){
        $this->db->update($this->get_table_name(). $this, ['nim'=>$this->nim]);
    }

    public function attribut_labels(){
        return['nim'=>'NIM:','nama'=>'Nama Mahasiswa:','prodi'=>'Prodi:'];
    }
}
```

entri.php

```php
<?php
class Entri extends CI_controller{
    public function index(){
        $this->load->model('Mahasiswa_model');
        $model = $this->Mahasiswa_model;

        if(isset($_POST['btnSubmit'])){
            $model->nim = $_POST['nim'];
            $model->nim = $_POST['nama'];
            $model->nim = $_POST['prodi'];
            $model->insert();
            $this->load->view('entri_respon_view',['model'=>$model]);
        }else{
            $this->load->view('entri_form_view',['model'=>$model]);
        }
    }
}
```

entri_from_view.php

```php
<html>
<head>
<title>Demo Model</title>
</head>
<body>
<h2>Entri Data Sukses</h2>

<form action="http://localhost/web2/index.php/entri" method="post">
<?php echo $model->labels['nim']; ?><br>
<input type="text" name="nim"><br><br>

<?php echo $model->labels['nama']; ?><br>
<input type="text" name="nama"><br><br>

<?php echo $model->labels['prodi']; ?><br>
<input type="text" name="prodi"><br><br>

<input type="submit" name="btnSubmit" value="Simpan">
</form>
</body>
</html>
```

entri_respon_view.php

```
<html>
<head>
<title>Demo Model</title>
</head>
<body>
<h2>Entri Data Sukses</h2>
<p>Bars data berikut telah ditambahkan ke tabel</p>

<table border="1">
<tr>
    <th>Kode</th>
    <th>Nama Produk</th>
    <th>Harga</th>
</tr>
<tr>
    <td><?php echo $model->nim; ?></td>
    <td><?php echo $model->nama; ?></td>
    <td><?php echo $model->prodi; ?></td>
</tr>
</body>
</html>
```

5. Latihan

a. Buatlah aplikasi untuk menghitung luas dan keliling tabung

b. Buatlah aplikasi untuk menghitung luas dan keliling tabung dimana nilai radius dan tingginyadidapatkan lewat inputan.

Matakuliah **:** Pemrograman Web Framework

MingguKe **:** 10

Waktu **:** 2 x 50 menit

Tema **:** Penanganan Form

1. **Kompetensi Dasar**
   a. Mahasiswa mampu memahami konsepdasarpenanganan form pada CI
   b. Mahasiswa mampu mempraktrikkanpenanganan form pada CI
   c. Mahasiswa mampu memberikan contoh penanganan form pada CI
2. **Dasar Teori**
   -
3. **AlatdanBahan**
   Laptop yang sudahterinstal:
   a. **Xampp**
   b. **Editor**
   c. **CI**
4. **Pembuatan form standart**

4.1. Kalkulator_model.php

```php
class Kalkulator_model extends CI_Model{
    public $var1;
    public $var2;
    public $operator;

    public $label = [];

    public function __construct(){
        parent::__construct();
        $this->operator = ['+','-','x','-','^'];
        $this->labels = $this->_attribut_labels();
    }

    public function hitung(){
        $hasil = NULL;
        switch ($this->operator){
            case '+':$hasil = $this->var1 + $this->var2; break;
            case '-':$hasil = $this->var1 - $this->var2; break;
            case '*':$hasil = $this->var1 * $this->var2; break;
            case ':':$hasil = $this->var1 / $this->var2; break;
            case '^':$hasil = pow($this->var1 , $this->var2); break;
        }
        return $hasil;
    }

    private function _attribut_labels(){
        return['var1'=>'Bilangan ke-1:',
        'var2'=>'Bilangan ke-2:',
        'operator'=>'Operasi :',];
    }
}
```

4.2. Kalkulator_form_view.php

```php
<html>
<head>
 <title>Demo Form</title>
</head>
<body>
<form action="http://localhost/web2/index.php/kalkulator/index" method="POST">
    <?php echo $model->labels['var1']; ?><br/>
    <input type="text" name="var1"><br/><br/>

    <?php echo $model->labels['operator']; ?><br/>
    <select name="operator" size="1">
    <option value="+">Tambah</option>
    <option value="-">Kurang</option>
    <option value="x">Kali</option>
    <option value="/">Bagi</option>
    <option value="^">Pangkat</option>
    </select><br/><br/>
    <?php echo $model->labels['var2']; ?><br/>
    <input type="text" name="var2"><br/><br/>
    <input type="submit" name="btnSubmit" value="Hitung" />
    </form>
</body>
</html>
```

4.3. Kalkulator_respon_view.php

```
<html>
<head>
 <title>Demo Form</title>
</head>
<body>
     <?php
     printf("%f %s %f = %f",
     $model->var1,
     $model->operator,
     $model->var2,
     $model->hitung());
     ?>
</body>
</html>
```

4.4. Kalkulator.php

```
<?php
class Kalkulator extends CI_Controller{
    private $model = NULL;

    public function __construct(){
        parent::__construct();
        $this->load->model('Kalkulator_model');
        $this->model = $this->Kalkulator_model;
    }

    public function index(){
        if(isset($_POST['btnSubmit'])){
            $this->model->var1 = $_POST['var1'];
            $this->model->var2 = $_POST['var2'];
            $this->model->operator = $_POST['operator'];
            $this->load->view('kalkulator_respon_view',['model'=>$this->model]);
        }else{
            $this->load->view('kalkulator_form_view',['model'=>$this->model]);
        }
    }
}
```

5. Pembuatan Form dengan helper

5.1. kalkulator1.php

```php
<?php
class Kalkulator1 extends CI_Controller{
    private $model = NULL;

    public function __construct(){
        parent::__construct();

        $this->load->helper('form');

        $this->load->model('Kalkulator_model');
        $this->model = $this->Kalkulator_model;
    }

    public function index(){
        if(isset($_POST['btnSubmit'])){
            $this->model->var1 = $_POST['var1'];
            $this->model->var2 = $_POST['var2'];
            $this->model->operator = $_POST['operator'];
            $this->load->view('kalkulator1_respon_view',['model'=>$this->model]);
        }else{
            $this->load->view('kalkulator1_form_view',['model'=>$this->model]);
        }
    }
}
```

5.2. kalkulator1_form_view

```php
<html>
<head>
 <title>Demo Form dengan helper</title>
</head>
<body>
<?php
    echo form_open('kalkulator1/index');
    echo form_label($model->labels['var1']);
    echo '</br>';
    echo form_input('var1');
    echo '<br/><br/>';

    echo form_label($model->labels['operator']);
    echo '<br />';
    $opsi = ['+'=>'Tambah','-'=>'kurang','*'=>'kali','/'=>'bagi','^'=>'pangkat'];
    echo form_dropdown('operator',$opsi);
    echo '<br/><br/>';

    echo form_label($model->labels['var2']);
    echo '</br>';
    echo form_input ('var2');
    echo '<br/><br/>';

    echo form_submit('btnSubmit','Hitung');
    echo form_close();

?>
</body>
</html>
```

5.3. kalkulator1_form_respon

```
1  <html>
2  <head>
3  <title>Demo Form</title>
4  </head>
5  <body>
6      <?php
7      printf("%.1f %s %.1f = %.1f",
8      $model->var1,
9      $model->operator,
10     $model->var2,
11     $model->hitung());
12     ?>
13 </body>
14 </html>
```

6. Tugas

a. Buatlah sebuah aplikasi untuk perhitungan persegi, persegipanjang, dan egitiga diamana ada inputan untuk angka 1 dan 2 kemudian pilihan dari list box untuk memilih menghitung persegi atau persegi panjang atau segitiga.

Matakuliah **:** Workshop Sistem Infromasi Terdistribusi

Minggu Ke **:** 11

Waktu **:** 2 x 50 menit

Tema **:** CRUD (Create Read Update Delete)

1. Kompetensi Dasar

a. Mahasiswa mampu memahamikonsep CRUD.

b. Mahasiswa mampu menerapkankonsep CRUD.

c. Mahasiswa mampu mengembangkankonsep CRUD pada sebuah studi kasus.

2. Dasar Teori

-

3. AlatdanBahan

Laptop yang sudahterinstal:

a. Xampp

b. Editor

c. Code Igniter

4. KegiatanPraktikum

a. Database

Tabel :Barang

No	NamaKolom	Type Data
1	Kode	Char / Varchar / String
2	Nama	Char / Varchar / String
3	Harga	Desimal / Float / Double
4	Stok	Int

b. Model

a) Buatlah sebuah Model dengan nama Barang_model.php

b) Lengkapi Program dibawah ini

```php
<?php
    class Barang_model extends CI_Model{

        //Buat 4 variabel dengan tipe public dengan nama seperti nama kolom di tabel barang
        //var 1
        //var 2
        //var 3
        //var 4
        public $label = [];

        public function __construct(){
            parent::__construct();
            $this->labels = $this->_attributeLabels();
            $this->load->database(); //fungsi untuk memuat database (library)
        }

        public function insert() {
            $sql = sprintf("INSERT INTO barang VALUES ('%s','%s','%f','%d')",
                    $this->kode,
                    ...........,  //5
                    ...........,  //6
                    ...........); //7
            $this->db->query($sql);
        }

        public function update() {
            $sql = sprintf("UPDATE `barang` SET `nama`='%s',harga=%f,`stok`=%d WHERE `kode`='%s'",
                    $this->kode,
                    ...........,  //8
                    ...........,  //9
                    ...........); //10
            $this->db->query($sql);
        }
        public function delete() {
            $sql = sprintf("DELETE FROM barang WHERE kode = '%s'", $this->kode);
            $this->db->query($sql);
        }

        public function read() {
            $sql = "SELECT * FROM barang ORDER BY kode";
            $query = $this->db->query($sql);
            return $query->result();
        }

        private function _attributeLabels() {
            return [
                'kode'=>'Kode: ',
                'nama'=>'Nama: ',
                'harga'=>'Harga: ',
                'stok'=>'Stok: '
            ];
        }
    }
?>
```

c. Controller

i. Buatlahsebuah Controller dengannamaCrud.php.

ii. Lengkapi Program dibawahini.

```php
<?php
class Crud extends CI_Controller {

    public $model = NULL;

    public function __construct() {
        parent::__construct();
        // load model, gunakan fungsi
        //$this->load->model('nama model') .........1
        //$this->model = $this->nama model .........2

        $this->load->database();
        $this->load->helper('url'); // sebagai redirect
    }

    public function index() {
        //gunakan $this untuk menrefer atau menunjuk pada function read() .........3
    }

    public function create() {
        if(isset($_POST['----nama tombol----'])) {
            $this->model->kode = $_POST['kode'];
            ..................................... //4
            ..................................... //5
            ..................................... //6
            $this->model->insert();
            redirect('crud');
        }else{
            $this->load->view('CRUD/crud_create_view', ['model'=>$this->model]);
        }
    }

    public function read() {
        $rows = $this->model->read();
        $this->load->view('CRUD/crud_read_view', ['rows'=>$rows]);
    }

    public function update($kode_up) {
        if(isset($_POST['btnSubmit'])) {
            $this->model->kode = $_POST['kode'];
            ..................................... //7
            ..................................... //8
            ..................................... //9
            $this->model->update();
            redirect('crud');
        }else{
            $query = $this->db->query("SELECT * FROM barang WHERE kode='$kode_up'");
            $row = $query->row();
            $this->model->kode = $row->kode;
            ..................................... //9
            ..................................... //10
            ..................................... //11
            $this->load->view('CRUD/crud_update_view', ['model'=>$this->model]);
        }
    }

    public function delete($kode_del) {
        $this->model->kode = $kode_del;
        $this->model->delete();
        redirect('crud');
    }
}
```

d. View

i. Buatlahsebuah folder didalam view dengannama CRUD

ii. BuatlahsebuahViewdidalam folder CRUD dengannama1.) *crud_read_view*, 2.) *crud _update_view*, 3.) *crud_create_view*

iii. Lengkapi Program dibawahini.

crud_read_view.php

```
<html>
<head>
    <title>Demo CRUD</title>
</head>
<body>
    <h2>Demo CRUD</h2>
    <p><a href="crud/create">Tambah Data</a></p>
    <table border="1">
        <tr>
            <th width="100">Kode</th>
            <th width="120">Nama</th>
            <th width="100">Harga</th>
            <th width="100">Stok</th>
            <th width="100"></th>
        </tr>

        <?php
            foreach($rows as $row) {
        ?>
        <tr>
            <td><?php echo $row->kode; ?></td>
            .....<!-- 1 -->
            .....<!-- 2 -->
            .....<!-- 3 -->
            <td align="center"><a href="crud/update/<?php echo $row->kode; ?>">Ubah</a>
            <a href="delete/<?php echo $row->kode; ?>">Hapus</a></td>
        </tr>
        <?php
            }
        ?>
    </table>
</body>
</html>
```

crud_create_view.php

```php
<html>
<head>
    <title>Demo CRUD</title>
</head>
<body>
    <h2>Demo CRUD</h2>
    <p><strong>Tambah Data</strong></p>

    <form action="create" method="post">
        <?php echo $model->labels['kode']; ?><br/>
        <input type="text" name="kode" size="10" maxlength="10"/><br/><br/>

        <?php echo $model->labels['nama']; ?><br/>
        <input type="text" name="nama" size="30" maxlength="25"/><br/><br/>

        <?php echo $model->labels['harga']; ?><br/>
        <textarea name="harga"></textarea><br/><br/>

        <?php echo $model->labels['stok']; ?><br/>
        <textarea name="stok"></textarea><br/><br/>

        <input type="submit" name="btnSubmit" value="Simpan"/>
        <input type="button" value="Batal" onclick="javascript:history.go(-1);"/>
    </form>
</body>
</html>
```

crud_update_view.php

```php
<html>
<head>
    <title>Demo CRUD</title>
</head>
<body>
    <h2>Demo CRUD</h2>
    <p><strong>Ubah Data</strong></p>

    <form action="create" method="POST">
        <?php echo $model->labels['kode']; ?><br/>
        <input type="text" name="kode" size="10" maxlength="10" value="<?php echo $model->kode; ?>"/><br/><br/>

        <?php echo $model->labels['nama']; ?><br/>
        <input type="text" name="nama" size="30" maxlength="25" value="<?php echo $model->nama; ?>"/><br/><br/>

        <?php echo $model->labels['harga']; ?><br/>
        <textarea name="harga"><?php echo $model->harga; ?></textarea><br/><br/>

        <?php echo $model->labels['stok']; ?><br/>
        <textarea name="stok"><?php echo $model->stok; ?></textarea><br/><br/>

        <input type="submit" name="btnSubmit" value="Simpan"/>
        <input type="button" value="Batal" onclick="javascript:history.go(-1);"/>
    </form>
</body>
</html>
```

5. Tugas

a) Lengkapi Program pada kegiatan praktikum.

b) Pahami fungsi setiap baris program.

c) Tambahkan fungsi search pada program diatas.

Matakuliah : Pemrograman Web Framework

Minggu Ke : 12

Waktu : 2 x 50 menit

Tema : Session & Cookies

1. Kompetensi Dasar

a. Mahasiswa mampu memahami konsep Session & Cookies.

b. Mahasiswa mampu menerapkan konsep Session & Cookies.

c. Mahasiswa mampu mengembangkan konsep Session & Cookies pada sebuah studi kasus.

2. Dasar Teori

-

3. AlatdanBahan

Laptop yang sudahterinstal:

a. Xampp

b. Editor

c. Code Igniter

4. KegiatanPraktikum

a. Cookies

Demo_Cookie.php

```php
<?php
class Demo_cookie extends CI_Controller{
    public function __construct(){
        parent::__construct();
        $this->load->helper('cookie');
    }

    public function index(){
        //membuat cookie
        set_cookie('myvar', 100, 60000 * 30);
        //load view
        $this->load->view('cookie_view');
    }
}
?>
```

Cookie_view.php

```
<html>
<head>
    <title>Demo Cookie</title>
</head>
<body>
<h2>Demo Cookie</h2>
<p><?php echo get_cookie('myvar'); ?></p>

</body>
</html>
```

b. Session

buka file config/autoload.php

```
$autoload['libraries'] = array('session');
```

Demo_Session.php

```
<?php
class Demo_session extends CI_Controller{
    public function __construct(){
        parent::__construct();
        $this->load->library('session');
    }

    public function index(){
        //membuat session
        $this>session->set_userdata('username','admin');
        //load view
        $this->load->view('session_view');
    }

    public function halaman1(){
        $this->load->view('session_view2');
    }

    public function halaman2(){
        $this->load->view('session_view3');
    }
}
?>
```

session_view.php

```
<html>
<head>
    <title>Demo Session</title>
</head>
<body>
<h2>Demo Session</h2>
<p>
<a href="http://localhost\web2\index.php\demo_session\halaman1">Halaman 1</a> 
<a href="http://localhost\web2\index.php\demo_session\halaman2">Halaman 2</a> 
</p>

</body>
</html>
```

session_view2.php

```
<html>
<head>
    <title>Demo Session</title>
</head>
<body>
<h2>Demo Session Halaman 1</h2>
<p>
<a href="http://localhost\web2\index.php\demo_session\halaman1">Halaman 1</a> 
<a href="http://localhost\web2\index.php\demo_session\halaman2">Halaman 2</a> 
</p>

<p>Username : <?php echo $this->session->userdata['username']; ?></p>

</body>
</html>
```

session_view3.php

```
<html>
<head>
    <title>Demo Session</title>
</head>
<body>
<h2>Demo Session Halaman 2</h2>
<p>
<a href="http://localhost\web2\index.php\demo_session\halaman1">Halaman 1</a> 
<a href="http://localhost\web2\index.php\demo_session\halaman2">Halaman 2</a> 
</p>

<p>Username : <?php echo $this->session->userdata['username']; ?></p>

</body>
</html>
```

c. Database

Tabel :User

No	NamaKolom	Type Data
1	user	Char / Varchar / String
2	password	Char / Varchar / String

Masukan 2 user kedalamtabel missal : 1. User : admin, password : 123456; 2. User : user1, password : 123456

Lalubuka file config/autoload.php

```
63  $autoload['libraries'] = array('session');
```

d. Model

a) Buatlah sebuah Model dengan nama Login_model.php

```
1  <?php
2  class Login_model extends CI_Model{
3      public $user;
4      public $password;
5
6      public $label = [];
7
8      public function __construct(){
9          parent::__construct();
10         $this->labels = $this->_attributeLabels();
11         $this->load->database(); //memuat database (library)
12     }
13
14     public function cek_log(){
15         $sql = sprintf("SELECT COUNT(*) AS hitung FROM `user` WHERE user='%s' AN
16                         password='%s'",
17         $this->user,
18         $this->password);
19         $query = $this->db->query($sql);
20         $row = $query->row_array();
21         return $row['cnt'] == 1;
22     }
23
24     private function _attributeLabels() {
25         return [
26         'user'=>'User :',
27         'password'=>'Password :'
28         ];
29     }
30 }
31 ?>
```

e. Controller

iii. Buatlahsebuah Controller dengannamaLogin.php.

```php
<?php
class Login extends CI_Controller{
    public $model = NULL;

    public function __construct(){
        parent::__construct();
        $this->load->model('Login_model');
        $this->model = $this->Login_model;

        $this->load->library('session');
        $this->load->helper('url');
    }

    public function index(){
        if (isset($_POST['btn_log'])){
            $this->model->user = $_POST['txt_user'];
            $this->model->password = $_POST['txt_password'];
            if ($this->model->cek_log()== TRUE){
                $this->session->set_userdata('user', $this->model->user);
                $this->load->view('Login/login_succes', ['model'=>$this->model]);
            }else{
                redirect('login');
            }
        }else{
        $this->load->view('Login/login_view', ['model'=>$this->model]);}
    }

    public function logout(){
        if($this->session->has_userdata('username')){
            $this->session->sess_destroy();
            $this->load->view('login_view',['model'=>$this->model]);
        }
    }
}
?>
```

f. View

i. Buatlahsebuah folder didalam view dengannamaLogin

ii. BuatlahsebuahViewdidalam folder CRUD dengannama1.) *login_view*, 2.) *login_succes*

1.*login_view.php*

```
<html>
<head>
    <title>Demo Session & Cookies</title>
</head>
<body>
    <h2 align="center">Login Demo</h2>
    <form action="login" method="POST" align="center">
        <b>Username</b><br>
        <input type="text" name="txt_user"><br><br>
        <b>Password</b><br>
        <input type="password" name="txt_pass"><br><br>
        <input type="submit" value="Login" name="btn_log">
    </form>
</body>
```

2. *login_succes.php*

```
<html>
<head>
    <title>Login</title>
</head>
<body>
<h2>Login Berhasil</h2>
<p>Selamat datang
<?php echo $model->user?></p>
<p>[<a href="http://localhost/web2/index.php/login/logout"> Logout </a>]</p>
</body>
</html>
```

5. Tugas

a) Pahami fungsi setiap baris program.

b) Gabungkan fungsi session dengan CRUD.Sehingga ketika berhasi login yang tampil adalah halaman berisikan table barang seperti pada modul 11

Matakuliah **:** Pemrograman Web Framework
Minggu Ke **:** 13-16
Waktu **:** 2 x 50 menit
Tema **:** Proyek Akhir Pemrograman Web Framework

1. Kompetensi Dasar

a. Mahasiswa mampu menemukan studi kasus permasalahan.
b. Mahasiswa mampu menerapkan konsep sistem informasi dalam menyelesaikan studi kasus.
c. Mahasiswa mampu mengembangkan sistem informasi sesuai dengan studi kasus.

2. Dasar Teori

-

3. Alat dan Bahan

Laptop yang sudah terinstal:

a. Xampp
b. Editor
c. Code Igniter

4. Kegiatan Praktikum

a) Buatlah kelompok beranggotakan 4-5 orang mahasiswa/i.
b) Setiap kelompok membangun sebuah website dengan ketentuan :
   i. Studi Kasus disamakan dengan Workshop aplikasi mobile dan terintegrasi.
   ii. Diwajibkan dibangun menggunakan code igniter.
   iii. Diperbolehkan menggunakan template.
   iv. Diwajibkan menggunakan CSS atau Bootsrap.
   v. Diperbolehkan menggunakan JQuery atau AJAX.
   vi. Wajib terdapat fungsi CRUD (Create Read Update Delete).
   vii. Wajib terdapat minimal 3 tabel di database.
   viii. Wajib terdapat session / cookies.
c) Penilain projek akhir berupa ujian lisan dan presentasi.

d) Update proyek wajib melalui Git (boleh via app Github, Source Tree dll.), masing-masing kelompok wajib menginvite dosen pengampu workshop.