PEMROGRAMAN INTERNET

MUHAMMAD HENDRA SUNARYA
MUHAMMAD BAHIT

Diterbitkan Atas Kerjasama
Deepublish dengan Politeknik Banjarmasin

PEMROGRAMAN INTERNET

UU No 28 tahun 2014 tentang Hak Cipta

Fungsi dan sifat hak cipta Pasal 4

Hak Cipta sebagaimana dimaksud dalam Pasal 3 huruf a merupakan hak eksklusif yang terdiri atas hak moral dan hak ekonomi.

Pembatasan Pelindungan Pasal 26

Ketentuan sebagaimana dimaksud dalam Pasal 23, Pasal 24, dan Pasal 25 tidak berlaku terhadap:

i. Penggunaan kutipan singkat Ciptaan dan/atau produk Hak Terkait untuk pelaporan peristiwa aktual yang ditujukan hanya untuk keperluan penyediaan informasi aktual;
ii. Penggandaan Ciptaan dan/atau produk Hak Terkait hanya untuk kepentingan penelitian ilmu pengetahuan;
iii. Penggandaan Ciptaan dan/atau produk Hak Terkait hanya untuk keperluan pengajaran, kecuali pertunjukan dan Fonogram yang telah dilakukan Pengumuman sebagai bahan ajar; dan
iv. Penggunaan untuk kepentingan pendidikan dan pengembangan ilmu pengetahuan yang memungkinkan suatu Ciptaan dan/atau produk Hak Terkait dapat digunakan tanpa izin Pelaku Pertunjukan, Produser Fonogram, atau Lembaga Penyiaran.

Sanksi Pelanggaran Pasal 113

1. Setiap Orang yang dengan tanpa hak melakukan pelanggaran hak ekonomi sebagaimana dimaksud dalam Pasal 9 ayat (1) huruf i untuk Penggunaan Secara Komersial dipidana dengan pidana penjara paling lama 1 (satu) tahun dan/atau pidana denda paling banyak Rp100.000.000 (seratus juta rupiah).
2. Setiap Orang yang dengan tanpa hak dan/atau tanpa izin Pencipta atau pemegang Hak Cipta melakukan pelanggaran hak ekonomi Pencipta sebagaimana dimaksud dalam Pasal 9 ayat (1) huruf c, huruf d, huruf f, dan/atau huruf h untuk Penggunaan Secara Komersial dipidana dengan pidana penjara paling lama 3 (tiga) tahun dan/atau pidana denda paling banyak Rp500.000.000,00 (lima ratus juta rupiah).

PEMROGRAMAN INTERNET

Muhammad Hendra Sunarya
Muhammad Bahit

PEMROGRAMAN INTERNET

Penulis :
Muhammad Hendra Sunarya & Muhammad Bahit

ISBN :
978-623-7694-08-3

ISBN Elektronis :
978-623-7694-33-5

Editor dan Penyunting :
Adi Pratomo

Desain Sampul dan Tata Letak :
Rahma Indera; Eko Sabar Prihatin

Penerbit :
POLIBAN PRESS
Anggota APPTI (Asosiasi Penerbit Perguruan Tinggi Indonesia)
no.004.098.1.06.2019
Cetakan Pertama, 2020

Redaksi :
Politeknik Negeri Banjarmasin, Jl. Brigjen H. Hasan Basry,
Pangeran, Komp. Kampus ULM, Banjarmasin Utara
Telp: (0511)3305052
Email: press@poliban.ac.id

Diterbitkan pertama kali oleh :
Poliban Press, Banjarmasin, Oktober 2020

Dicetak oleh :
PERCETAKAN DEEPUBLISH
Jl.Rajawali, G. Elang 6, No 3, Drono, Sardonoharjo, Ngaglik, Sleman
Jl.Kaliurang Km.9,3 – Yogyakarta 55581
Telp/Faks: (0274) 4533427
Website: www.deepublish.co.id
www.penerbitdeepublish.com
E-mail: cs@deepublish.co.id

Katalog Dalam Terbitan (KDT)
Muhammad Hendra Sunarya; Muhammad Bahit — Cet. **1**. — **Pemrograman Internet,**
Banjarmasin: Poliban Press, Oktober 2020.

xii; **60** hlm.; **15.5x23 cm**

UCAPAN TERIMA KASIH

Ucapan terima kasih kami sampaikan kepada Poliban Press karena telah mempercayakan proses percetakan buku *Pemrograman Internet* kepada Penerbit Deepublish. Semoga buku ini dapat memberikan manfaat kepada seluruh pembaca dan kerja sama ini dapat terus terjalin.

KATA PENGANTAR

Puji syukur ke hadirat Allah Swt. atas limpahan rahmat dan karunianya sehingga buku *Pemrograman Internet* tahun 2020 telah dapat diselesaikan. Buku ini merupakan pengantar bagi mahasiswa Diploma Komputer Akuntansi.

Terima kasih disampaikan kepada Joni Riadi S.S.T., M.T. selaku Direktur Politeknik Negeri Banjarmasin dan Nurmahaludin, S.T., M.T. selaku Ketua Pusat Penelitian dan Pengabdian Masyarakat beserta sekretaris dan staf. Terima kasih juga disampaikan kepada Faris Ade Irawan, Reza Fauzan, Eko Sabar Prihatin, dan Rahma Indera yang telah berkontribusi dalam editing serta seluruh tim Poliban Press dan semua pihak yang telah ikut membantu dalam penyelesaian buku ini.

Kami menyadari masih terdapat kekurangan dalam buku ini untuk itu kritik dan saran terhadap penyempurnaan buku ini sangat diharapkan. Semoga buku ini dapat memberi manfaat bagi semua pihak.

Banjarmasin, September 2020

Poliban Press

PRAKATA

Syukur alhamdulillah penulis panjatkan ke hadirat Allah Swt. karena atas rahmat dan rida-Nya jualah buku ajar ini dapat diselesaikan.

Buku ajar ini dibuat sebagai Kurikulum Politeknik Negeri Banjarmasin Jurusan Akuntansi, program studi Komputerisasi Akuntansi yang mulai diberlakukan pada tahun akademik 2020-2021.

Buku ajar ini mengulas tentang bahasa komputer yang digunakan untuk membuat sebuah program berbasis internet, dengan pendekatan praktik secara langsung, yang mana bahasa tersebut meliputi html, css, Bootstrap, php serta juga sedikit menjelaskan tentang penerapan Library Paris ORM.

Penulis menyadari bahwa modul ini sangatlah jauh dari kata sempurna sehingga penulis mengharapkan kritik dan saran yang membangun demi perbaikan buku ajar ini untuk masa yang akan datang.

Penulis mengharapkan semoga buku ajar ini dapat bermanfaat bagi semua pihak yang membutuhkannya.

Banjarmasin, 28 Mei 2020

Muhammad Hendra Sunarya, M.Sc.

DAFTAR ISI

UCAPAN TERIMA KASIH v
KATA PENGANTAR vi
PRAKATA vii
DAFTAR ISI viii
DAFTAR GAMBAR x
DAFTAR TABEL xii
BAB I HTML 1
1.1. Pengertian *Tag* HTML 1
1.2. Kumpulan *Tag* HTML 3
1.3. Soal latihan 9
BAB II CSS 10
2.1. Pengertian CSS 10
2.2. Kegunaan CSS 10
2.3. Contoh Penulisan CSS 11
2.4. Soal Latihan 15
BAB III BOOTSTRAP 16
3.1. Pengertian Bootstrap 16
3.2. Cara Instal Bootstrap 17
3.3. Sistem *Grid* pada Bootstrap 19
3.4. Gambar *Preview* pada Bootstrap 25
3.5. Soal Latihan 28
BAB IV BOOTSTRAP DAN CSS 29
4.1. Sistem *Grid* pada Bootstrap 29
4.2. Tipografi 33

4.3. Soal Latihan 33

BAB V PHP 34

5.1. Pengertian PHP 34
5.2. Kebutuhan Perangkat Lunak 34
5.3. Membuat Halaman *Web* Sederhana dengan PHP 35
5.4. Jenis-Jenis *Tag* PHP 35
5.5. Penggunaan Komentar pada PHP 37
5.6. Penulisan Karakter Khusus dengan tanda 37
5.7. Variabel PHP 37
5.8. Operator 38
5.9. Soal Latihan 38

BAB VI DATABASE 39

6.1. Pengertian *Database* 39
6.2. Database MySQL 39
6.3. Jenis-Jenis Tipe Data *Database* MySQL 40
6.4. Langkah Pembuatan *Database* MySQL 43
6.5. Penyederhanaan Struktur *Database* 47
6.6. Menentukan *Has One* 48
6.7. *Query* Paris ORM 50
6.8. Menentukan *Has Many* 50
6.9. Menentukan *Belongs to* 52
6.10. Soal Latihan 53

BAB VII PARIS ORM COMPOSER 54

7.1. Paris ORM *Composer* 54
7.2. Cara Install Paris ORM Composer 54
7.3. Membuat Format Tanggal dari Hari, Bulan, Tahun Dengan Menggunakan *Library Carbon* 57
7.4. Soal Latihan 58

DAFTAR PUSTAKA 60

DAFTAR GAMBAR

Gambar 1.1 Output Tag Dasar HTML 4
Gambar 1.2 *Output Tag* Tabel 8
Gambar 3.1 Contoh Versi Bootstrap 17
Gambar 3.2 *Download* Bootstrap 18
Gambar 3.3 Hasil *Extract* Zip Bootstrap 18
Gambar 3.4 *Output Container-fluid* 2 Kolom 20
Gambar 3.5 Output Container-fluid 2 Baris 21
Gambar 3.6 Hasil Output Bootstrap 22
Gambar 3.7 *Output* Bootstrap dengan CSS 23
Gambar 3.8 Output Kelas Container-fluid 24
Gambar 3.9 Tampilan Gambar dengan Bootstrap 26
Gambar 3.10 Tampilan Gambar *Circle* dengan Bootstrap 27
Gambar 3.11 Tampilan Gambar *Round* dengan Bootstrap 27
Gambar 6.1 *Login* phpmyadmin 44
Gambar 6.2 Setelah *Login* 44
Gambar 6.3 Membuat *Database* 45
Gambar 6.4 Membuat Tabel 45
Gambar 6.5 Membuat *Field* pada Tabel 46
Gambar 6.6 Data Pada Tabel 46
Gambar 6.7 Relasi *Database* 47

Gambar 6.8 Daftar Tabel 48
Gambar 6.9 Relasi Antar Tabel 50
Gambar 7.1 Download Composer 54
Gambar 7.2 Directory Penyimpanan *File* 55
Gambar 7.3 Hasil *Install Composer* 55
Gambar 7.4 Jendela *Command Prompt* 56
Gambar 7.5 *Output Install Composer* Selesai 57
Gambar 7.6 *Library* Carbon 57
Gambar 7.7 Penulisan *Script Library* Carbon 58

DAFTAR TABEL

Tabel 1.1 *Tag* Dasar HTML 3

Tabel 1.2 Daftar Tag *Text* dan *Heading* 4

Tabel 1.3 Daftar *Tag* Media 6

Tabel 1.4 *Tag* Tabel HTML 7

Tabel 3.1 Daftar Kelas Grib 19

Tabel 5.1 *Tag* PHP 36

Tabel 5.2 Kode Komentar pada PHP 37

Tabel 5.3 Karakter Khusus PHP 37

Tabel 5.4 Operator PHP 38

Tabel 5.5 Operator Pembanding 38

BAB I

HTML

Capaian Pembelajaran

1. Mampu memahami konsep dari HTML.
2. Mampu memahami struktur HTML.
3. Mampu memahami *tag* dasar HTML.
4. Mampu mengimplementasikan *tag* HTML.
5. Mampu menjalankan HTML ke dalam *web browser*.

1.1 Pengertian *Tag* HTML

Tag merupakan elemen awal yang ditulis pada HTML, di mana *tag* memiliki 1 buah *tag* pembuka dan *tag* penutup yang berfungsi untuk menjelaskan akhir dari sebuah *tag*. Contoh pemberian *tag* adalah <namatag> dan ditutup kembali dengan </namatag>**.** Untuk <Namatag> menandakan bahwa *code* tersebut akan membuka suatu *tag* sedangkan </namatag> menandakan bahwa *tag* tersebut telah berakhir, seperti yang telah dijelaskan pada halaman sebelumnya bahwa dalam HTML setiap *tag* yang sudah dibuka harus ditutup kembali supaya tidak mengganggu pada elemen *tag* yang lainnya. Selain itu juga beberapa *tag* yang tidak perlu untuk ditutup kembali seperti *tag* </br> <hr/>.

HTML (*Hypertext Markup Language*) merupakan bahasa markah yang digunakan untuk membuat sebuah halaman *web* dengan tujuan untuk menampilkan atau berbagai informasi dalam sebuah *web* yang akses melalui internet (Wempen, 2011). Semua halaman *web* yang sering anda buka, seperti facebook.com, Twitter.com, google.com akan ditampilkan menggunakan HTML. Jadi bisa dikatakan HTML adalah bahasa dasar untuk menampilkan halaman *web* pada *web browser* (Ariona, 2013).

HTML terdiri dari 3 komponen kata yaitu *Hypertext*, *Markup* dan *Language*. Kata *Hypertext* dari HTML berarti "***text***" yang tidak hanya berfungsi sebagai *text* biasa tetapi juga dapat berfungsi sebagai penghubung ke halaman lain atau yang sering dikenal dengan istilah *link*. Selain *text* dapat dijadikan *link*, sebuah gambar juga dapat dijadikan sebagai *link* untuk penghubung ke halaman lain (Pratama, 2016).

Kata kedua dari HTML adalah Markup yang diambil dari kata bahasa inggris yaitu *mark* yang artinya "**tanda atau penanda**". HTML menggunakan tanda-tanda khusus yang diperlukan untuk mengatur dan membuat struktur halaman *web* seperti tanda-tanda <p>, <a> dan masih banyak tanda-tanda khusus yang digunakan. Tanda-tanda ini pada HTML dikenal sebagai *tag* yang berfungsi memberikan informasi kepada *web browser* (Pratama, 2016).

Kata ketiga adalah *Language* yang berarti "**bahasa**". HTML tidak memiliki struktur dasar seperti variabel, kondisi IF, *function* atau *class* layaknya sebuah bahasa pemrograman komputer yang menggunakan *programming language*. Sehingga secara tidak langsung HTML dapat dikatakan bukanlah bahasa pemrograman (Pratama, 2016).

Dapat disimpulkan bahwa HTML (*hypertext markup language*) adalah sekumpulan simbol-simbol atau *tag-tag* yang dituliskan dalam sebuah *file* dengan tujuan untuk menampilkan informasi dari yang dituliskan dalam bentuk simbol atau tersebut pada halaman *web browser* (Duckett, 2011). *Tag* tersebut untuk memberikan informasi kepada *browser* untuk menampilkan halaman *web* dengan lengkap kepada pengguna.

Tag HTML selalu diawali dengan *tag* pembuka <x> dan *tag* penutup </x> (Duckett, 2011). Sebuah halaman *website* diapit oleh *tag* pembuka <html> dan *tag* penutup /html> untuk memulai menuliskan perintah-perintah HTML. *File* HTML selalu berakhiran dengan ekstensi *.htm atau *.html. Jadi jika anda mengetik sebuah naskah dan menyimpannya dengan ekstensi *.html maka anda membuat *file* yang berformat HTML.

1.2 Kumpulan *Tag* HTML

Setiap *tag* pada bahasa html ada yang disebut *tag* pembuka dan *tag* penutup. Mereka harus ada bersamaan di sisi luar dari isi. Contohnya kita memberikan judul dengan *tag h1*, maka tulisan judul berada di antara *tag h1* pembuka dan *tag h1* penutup artinya *tag h1* pembuka dan penutup berada di sisi luar dari isi.

Bagaimana membedakan *tag* pembuka dan *tag* penutup? *Tag* pembuka adalah tulisan normal tanpa tambahan karakter apapun di dalam *tag*, sedangkan *tag* penutup ditandai dengan garis miring (/) sebelum penulisan jenis *tag*. Bisa dilihat pada gambar diatas di akhirnya ada </h1> artinya *tag* h1 diberi garis miring (/) sebelum penulisannya maka ini adalah *tag* penutup. Mari kita kenali lebih jauh lagi beberapa *tag* html yang ada.

1. *Tag* Dasar HTML

Tabel 1.1 *Tag* Dasar HTML

Tag	**Keterangan**
!DOCTYPE html	Merupakan deklarasi dari html 5 & penulisan deklarasi dalam huruf besar ataupun kecil tidak menjadi masalah.
<html></html>	Digunakan untuk membuat sebuah dokumen html.
<head> </head>	Memberikan informasi pada sebuah dokumen tersebut, pada pembuatan *website* biasa digunakan untuk insert *file* seperti CSS
<title> </title>	Memberikan judul pada dokumen html tersebut
<body></body>	Menggambarkan isi dari halaman atau dokumen html tersebut
<header> </header>	Digunakan untuk menampilkan tajuk kepala atau bagian *header* (atas), *tag* ini hanya bisa digunakan pada html 5. Secara *default* jika tidak diatur, maka *browser* akan menampilkan bagian *header* ini berupa blok
<footer> </footer>	Digunakan untuk memberikan bagian kaki pada sebuah dokumen html, biasanya digunakan untuk informasi atau hak cipta

Sekarang tuliskan kode HTML di bawah ini menggunakan *text editor* yang anda gunakan.

```
<html>
<head><title>Belajar Tag HTML</title></head>
<body>
   Disini adalah konten yang tampil di browser
</body>
</html>
```

Selanjutnya buka *file* tersebut dengan *browser* internet yang digunakan maka hasilnya dapat dilihat seperti Gambar 1.1 di bawah ini.

Gambar 1.1 Output Tag Dasar HTML

Tabel 1.2 Daftar Tag *Text* dan *Heading*

Tag	Keterangan
	Digunakan untuk *heading* atau judul dan biasa diartikan sebagai *text* yang sangat penting karena h1 ini memiliki ukuran yang sangat besar dibanding *tag* sejenisnya
	Merupakan tulisan penting atau judul kedua setelah kakaknya yaitu h1, ukurannya lebih kecil dari h1
	Merupakan tulisan penting atau judul ketiga setelah kakaknya yaitu h2, ukurannya lebih kecil dari h2
	Merupakan tulisan penting atau judul keempat setelah kakaknya yaitu h3, ukurannya lebih kecil dari h3
	Merupakan tulisan penting atau judul kelima setelah kakaknya yaitu h4, ukurannya lebih kecil dari h4
	Merupakan tulisan penting atau judul keenam setelah kakaknya yaitu h5, ukurannya lebih kecil dari h5
	Digunakan untuk membuat sebuah paragraf
	Merupakan *tag* frasa, digunakan untuk mempertebal tulisan biasa digunakan untuk tulisan penting pada paragraf
	Sama seperti *strong*, menjadi lebih tebal, biasa digunakan untuk *text* yang penting

Tag	Keterangan
	Digunakan pada teks yang bersifat menekankan, *tag* ini bertekstur miring
	Sama seperti em yang memiringkan tulisan
<mark> </mark>	Digunakan untuk memberi tanda pada tulisan (highlight), seperti kalian mencoret beberapa *text* menggunakan stabilo pada lembaran buku. Ini hanya berlaku ketika digunakan di html 5
<code> </code>	Digunakan untuk menampilkan beberapa kode komputer
	Digunakan untuk membuat *text* berukuran kecil, ini biasa digunakan ketika ingin memasukkan tulisan kecil di antara *text* yang berukuran besar misalnya h1 / h2, dll
<u> </u>	Digunakan untuk memberikan garis bawah pada *text*
<ins> </ins>	Digunakan untuk memberikan garis bawah pada *text*, biasa digunakan untuk *text* yang disisipkan
	Digunakan untuk membuat *text* sedikit lebih tinggi posisinya, biasa digunakan untuk angka atau pangkat pada matematika
	Digunakan untuk membuat *text* sedikit lebih tinggi posisinya, biasa digunakan untuk angka atau pangkat pada matematika
<samp> </samp>	Merupakan *tag* frasa, digunakan untuk menampilkan contoh *output* dari program komputer. *Tag* ini kaku layaknya bahasa program untuk membuat halaman *website* lebih mudah digunakan menggunakan CSS
<del> </del>	Digunakan untuk menampilkan garis tengah pada *text*, bermaksud untuk memberi tanda bahwa *file* pada tulisan tersebut sudah dihapus
<strike> </strike>	Digunakan untuk menampilkan garis tengah pada *text*, bermaksud untuk memberi tanda bahwa *file* pada tulisan tersebut sudah dihapus
<kbd> </kbd>	Digunakan untuk memberikan *text* seperti input keyboard. Akan tetapi *text* ini terlihat kaku, perlu diperbaiki lagi menggunakan CSS
<var> </var>	Digunakan untuk membuat *text* variabel, *text* ini bisa lebih menarik lagi jika dihias menggunakan CSS

2. *Text* Format dan *Heading*

Text formatting digunakan untuk melakukan pengaturan atau konfigurasi pada teks yang dijadikan isi dari sebuah halaman *web*. Pengaturan *text* terlihat secara real pada halaman *web* yang diinputkan

pada dokumen HTML. Tujuan penggunaan *text formatting* untuk membuat tampilan teks pada halaman *web* menjadi lebih baik atau mudah dipahami oleh pengguna. Secara umum pengaturan teks *formatting* meliputi pengaturan tebal, miring, garis bawah, teks berpangkat (Robbins, 2012).

Tag heading digunakan untuk membuat judul pada halaman *web*, *tag heading* pada HTML terdiri dari 6 level yaitu h1, h2, h3, h4, h5, h6. *Tag heading* secara *default* di tampilkan oleh *browser* dengan huruf tebal, *tag heading* terbesar adalah h1 dan terkecil adalah h6 (Pratama, 2016).

3. Media (Gambar, Audio dan Video)

Seperti yang kita ketahui, dalam bahasa pemrograman HTML dapat memasukkan atau menyisipkan media baik berupa gambar, audio bahkan video.

Tabel 1.3 Daftar *Tag* Media

Tag	Keterangan
<img src=””>	Digunakan untuk memasukkan gambar, sedangkan src adalah nama file gambar yang akan dimasukkan (pastikan kalian juga memasukkan urutan letak gambar tersebut
<audio controls> </audio>	Digunakan untuk memasukkan audio, sedangkan *controls* adalah pengendalian pada audio tersebut seperti *play/pause/stop*
<source src=”” type= “audio/wav”>	Digunakan untuk menampilkan audio pada audio *player* (*tag* di atas), maka *tag* ini disisipkan di antara *tag* pembuka <audio> dan penutup </audio>. Src adalah nama *file* audio yang akan dimainkan (pastikan kalian juga menyertakan letak audio tersebut disimpan secara berurutan), sedangkan *type* adalah jenis dari audio tersebut misalnya .wav/.mp3
<video controls> </video>	Digunakan untuk menampilkan video, sedangkan *controls* untuk pengendalian seperti *play*/pause/stop. Di antara *tag* ini kalian masukkan *tag source* seperti pada audio di atas. Dan video kalian akan muncul sesuai ukuran asli video yang disisipkan di *source*, untuk mengatur ukuran lebar dan tinggi video silahkan berikan width=”” dan height=”” di dalam *tag* pembuka <video> (pastikan kalian menulis setelah tulisan video, maka akan seperti ini <video width=”” height=”” controls>
<iframe>	Digunakan untuk menampilkan video, ini sama dengan *tag* video

Tag	Keterangan
</iframe>	di atas. *Tag* ini lebih umum yang biasa digunakan pada media sosial untuk menyematkan (*embed*) *post*, video, dll. Untuk menampilkan video silahkan tambahkan src di dalam *tag* pembuka iframe setelah tulisan iframe maka akan seperti ini <iframe src=””>. Pada src masukkan video yang akan disisipkan seperti penjelasan *tag* video di atas, *default*-nya video akan tampil sesuai ukuran video. Maka atur width=”” dan height=”” dan simpan seperti penjelasan *tag* video data.

4. Tabel

Untuk menampilkan data yang terstruktur atau tampilan data dari *database* biasanya disajikan dalam bentuk tabel sehingga data yang disajikan mudah dipahami oleh pengguna (Wempen, 2011). Untuk menampilkan data dalam bentuk tabel pada HTML ada empat *tag* yang digunakan yaitu *tag* <table>, *tag* <th>, *tag* <tr>, dan *tag* <td> (Robbins, 2012).

Tabel 1.4 *Tag* Tabel HTML

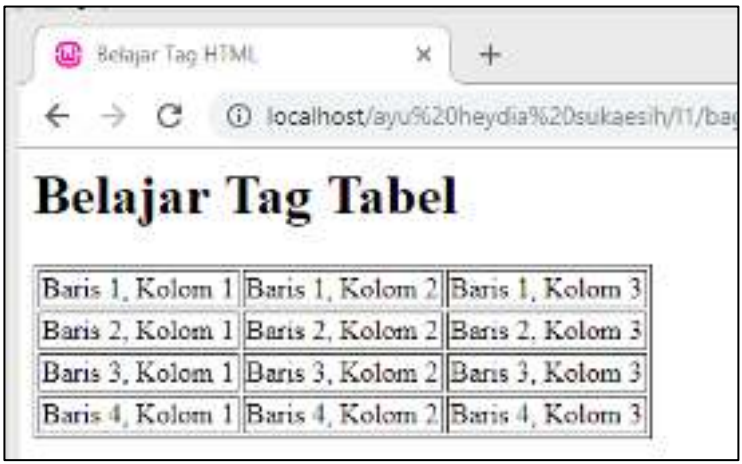
Tag	Keterangan
<table></table>	Mendefinisikan sebuah tabel
<tr></tr>	Mendefinisikan baris table
<th></th>	Mendefinisikan judul kolom
<td></td>	Mendefinisikan isi tiap kolom

Sekarang tuliskan kode HTML di bawah ini menggunakan *text editor*

```
<html>
  <head>
    <title>Belajar Tag HTML</title>
  </head>
  <body>
    <h1>Belajar Tag Tabel</h1>
    <table border="1">
      <tr>
      <td>Baris 1, Kolom 1</td>
      <td>Baris 1, Kolom 2</td>
```

```
      <td>Baris 1, Kolom 3</td>
      </tr>
      <tr>
      <td>Baris 2, Kolom 1</td>
      <td>Baris 2, Kolom 2</td>
      <td>Baris 2, Kolom 3</td>
      </tr>
      <tr>
      <td>Baris 3, Kolom 1</td>
      <td>Baris 3, Kolom 2</td>
      <td>Baris 3, Kolom 3</td>
      </tr>
      <tr>
        <td>Baris 4, Kolom 1</td>
        <td>Baris 4, Kolom 2</td>
        <td>Baris 4, Kolom 3</td>
      </tr>
    </table>
  </body>
</html>
```

Dari kode HTML diatas dapat dilihat pada Gambar 1.2 di bawah ini dengan *output* tiga kolom dan empat baris.



Gambar 1. 2 *Output Tag* Tabel

1.3 Soal latihan

1. Buatlah halaman *web* html dengan memuat tulisan minimal 500 kata disertai dengan *tag* html, lalu jalankan di *browser*!
2. Buatlah ringkasan tentang html ke dalam Google Docs, lengkap dengan contoh penggunaannya, dengan menggunakan bahasa sendiri!

BAB II

CSS

Capaian Pembelajaran

1. Mampu memahami konsep dari CSS.
2. Mampu memahami penggunaan CSS.
3. Mampu memahami penulisan kode CSS.
4. Mampu mengimplementasikan kode CSS ke dalam HTML.

2.1 Pengertian CSS

CSS adalah *Cascading Style Sheet* versi ke 3, yaitu pengatur dan pengendali tampilan sebuah halaman blog/ *web*. CSS3 melakukan penataan terhadap komponen HTML maupun XHTML pada halaman *web* sehingga menghasilkan tampilan yang ramah di mata atau retina *friendly*. CSS pada mulanya dipelopori dan dikembangkan serta distandardisasi oleh *World Wide Web Consortium* atau W3C pada tahun 1996 (Robbins, 2012).

2.2 Kegunaan CSS

CSS berguna untuk mengatur atau mengendalikan *border*, *line*, *font*, *background*, dan juga *content website* kita terlihat bagus. Dan sesuai perkembangan ada CSS1, CSS2, dan CSS3 (Howe, 2014). CSS3 mempunyai banyak kelebihan yang dibutuhkan *website*. Di CSS3 ini kita dapat melakukan animasi, mulai dari animasi warna sampai 3D. Dengan CSS3 desainer lebih dimudahkan dalam hal kompatibilitas *website*nya pada *smartphone* dengan dukungan fitur baru yakni media *query*.

2.3 Contoh Penulisan CSS

Penulisan kode CSS dalam HTML dibagi menjadi tiga cara, internal, *inline* dan eksternal. Pembagian ini berdasarkan letak kode CSS tersebut ditulis.

1. Internal CSS

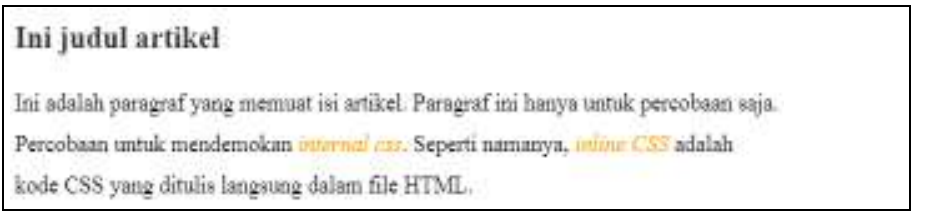
Internal CSS adalah kode CSS yang ditulis di *tag* <style>. Intarnal CSS juga dikenal dengan sebutan *Embeded CSS*. *Tag* <style> biasanya ditulis di dalam *tag* <head>. Bisa juga ditulis di dalam <body>, namun lebih banyak ditulis di dalam <head>. Contoh:

```
<!DOCTYPE html>
<html>
<head>
 <title>Contoh Internal CSS</title>
 <!-- penulisan internal css dalam tag head -->
 <style type="text/css">
 p{
 font-family: serif;
 line-height: 1.75em;
 font-size: 18px;
 }
 i {
 font-family: sans;
 color: orange;
 }
 </style>
</head>

<body>
 <!-- penulisan internal css dalam tag body -->
 <style type="text/css">
 h2 {
 font-family: sans;
 color: #333;
 }
```

```
 </style>
 <h2>Ini judul artikel</h2>
 <p>Ini adalah paragraf yang memuat isi artikel.
Paragraf ini hanya untuk percobaan saja. Percobaan
untuk mendemokan <i>internal css</i>. Seperti namanya,
<i>inline CSS</i> adalah kode CSS yang ditulis langsung
dalam file HTML.</p>
</body>
</html>
```

Hasilnya:



2. Eksternal CSS

Eksternal CSS adalah kode CSS yang ditulis terpisah dengan kode HTML. Eksternal CSS ditulis di sebuah *file* khusus yang berekstensi .css. Sebagai contoh, saya akan membuat sebuah *file* bernama style-ku.css. Berikut ini cuplikan isi *file* style-ku.css. Contoh:

```
p {
 font-family: serif;
 line-height: 1.75em;
}
i {
 font-family: sans;
 color: orange;
}
h2 {
 font-family: sans;
 color: #333;
}
```

Untuk menggunakan CSS tersebut dalam HTML, kita perlu mengimpornya. Ada beberapa cara memasukkan kode CSS dari berkas eksternal: Pertama menggunakan *tag* <*link*>.

```
<link rel="stylesheet" type="text/css" href="style-ku.css">
```

Atau bisa juga bisa menggunakan @import.

```
<style type="text/css">
@import "style-ku.css";
</style>
```

Penulisan pada HTML versi lengkapnya seperti ini.

```
<!DOCTYPE html>
<html>
<head>
 <title>Contoh Eksternal CSS</title>
 <link rel="stylesheet" type="text/css" href="style-ku.css">
</head>

<body>
 <h2>Ini judul artikel</h2>
 <p>Ini adalah paragraf yang memuat isi artikel. Paragraf ini hanya untuk percobaan saja. Percobaan untuk mendemokan <i>internal css</i>. Seperti namanya, <i>inline CSS</i> adalah kode CSS yang ditulis langsung dalam file HTML.</p>
</body>
</html>
```

Hasilnya pun akan sama seperti contoh internal CSS, karena kode CSS-nya sama. Hanya saja berbeda tempat penulisannya.

3. *Inline* CSS

Inline CSS adalah kode CSS yang ditulis langsung pada atribut elemen HTML. Setiap elemen HTML memiliki atribut *style*, di sana lah *inline* CSS ditulis. Contohnya seperti ini.

```
<h2 style="color:red; font-family: sans;">Ini judul artikel</h2>
```

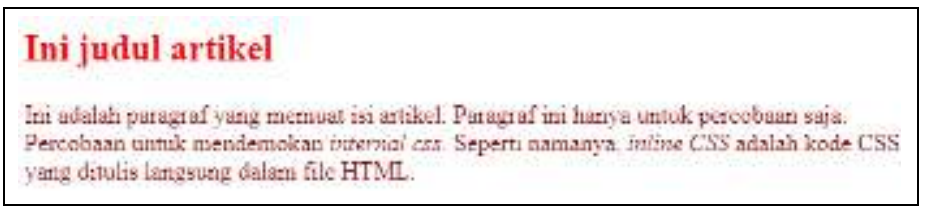
Contoh lengkap sebagai berikut.

```
<!DOCTYPE html>
<html>
<head>
 <title>Contoh Inline CSS</title>
</head>

<body>

<h2 style="color:red;font-family:sans">Ini judul artikel</h2>
 <p style="color:maroon">Ini adalah paragraf yang memuat isi artikel. Paragraf ini hanya untuk percobaan saja. Percobaan untuk mendemokan <i>internal css</i>. Seperti namanya, <i>inline CSS</i> adalah kode CSS yang ditulis langsung dalam file HTML.</p>
</body>
</html>
```

Hasilnya:



2.4 Soal Latihan

1. Tambahkanlah *script* css pada hasil pekerjaan bab 1 sebelumnya!
2. Buatlah satu buah *form input* dengan menggunakan *tag* tabel html, yang menyertakan *button* simpan dan batal dengan memilih salah satu contoh formulir di bawah ini:
   - Formulir pendaftaran
   - Formulir *input* data barang
3. Buatlah ringkasan tentang css ke dalam Google Docs, lengkap dengan contoh penggunaannya, dengan menggunakan bahasa sendiri!

BAB III

BOOTSTRAP

Capaian Pembelajaran

1. Mampu memahami konsep dari Bootstrap.
2. Mampu memahami kegunaan Bootstrap.
3. Mampu menginstal Bootstrap.
4. Mampu memahami pembagian dari Bootstrap.
5. Mampu memahami penulisan kode Bootstrap.
6. Mampu mengimplementasikan Bootstrap pada HTML serta CSS.

3.1 Pengertian Bootstrap

Bootstrap adalah sebuah *library framework* CSS yang dibuat khusus untuk bagian pengembangan *front-end website*. Bootstrap juga merupakan salah satu *framework* HTML, CSS dan Javascript yang paling populer di kalangan *web developer* yang digunakan untuk mengembangkan sebuah *website* yang *responsive*. Sehingga halaman *website* nantinya dapat menyesuaikan sesuai dengan ukuran monitor *device* (desktop, tablet, ponsel) yang digunakan pengguna di saat mengakses *website* dari *browser*. Pada mulanya Bootstrap bernama "Twitter Blueprint" yang dikembangkan oleh Mark Otto dan Jacob Thornton di Twitter sebagai kerangka kerja untuk mendorong konsistensi di alat internal.

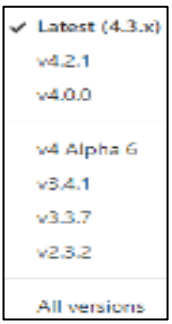
Dengan menggunakan Bootstrap seorang developer dapat dengan mudah dan cepat dalam membuat *front-end* sebuah *website*. Anda hanya perlu memanggil *class-class* yang diperlukan, misalnya membuat tombol, *grid*, tabel, navigasi dan lainnya. Bootstrap telah menyediakan kumpulan komponen *class interface* dasar yang telah dirancang sedemikian rupa untuk menciptakan sebuah tampilan yang menarik dan ringan. Selain

komponen *class interface*, Bootstrap juga memiliki *grid* yang berfungsi untuk mengatur *layout* pada halaman *website*. Selain itu *developer* juga dapat menambahkan *class* dan CSS sendiri, sehingga memungkinkan untuk membuat desain yang lebih variatif. Salah satu contoh *website* yang menggunakan *framework* Bootstrap yaitu Twitter. Bootstrap sendiri sebenarnya dikembangkan oleh *developer* Twitter sehingga Bootstrap sering juga disebut dengan "Twitter Bootstrap ". Bootstrap sendiri sudah kompatibel dengan versi terbaru dari beberapa *browser* seperti Google Chrome, Firefox, Internet Explorer, dan safari *browser*. Meskipun beberapa *browser* ini tidak didukung pada semua *platform*. Cukup banyak pengembang yang menggunakan Bootstrap dalam membuat *front-end website* karena beberapa kelebihan berikut.

- Dapat mempercepat waktu proses pembuatan *front-end website.*
- Tampilan Bootstrap yang sudah cukup terlihat modern.
- Tampilan Bootstrap sudah *responsive*, sehingga mendukung segala jenis resolusi, baik itu PC, tablet, dan juga smartphone.
- *Website* menjadi Sangat ringan ketika diakses, karena Bootstrap dibuat dengan sangat terstruktur.

3.2 Cara Instal Bootstrap

1. Kunjungi situs https://getBootstrap.com/.
2. Pilih versi yang diinginkan di menu atas kanan (misalnya v3.4.1).



Gambar 3.1 Contoh Versi Bootstrap

3. Klik tombol *Download*, akan diarahkan ke dokumentasi Bootstrap 3.

Gambar 3.2 *Download* Bootstrap

4. Pilihlah jenis yang diinginkan apakah *Compiled* CSS dan JS atau *Source File*s.
5. Klik tombol *Download* pada jenis yang diinginkan, tunggu sampai proses *download* selesai, buka hasil *download* kemudian *extract* ZIP maka akan muncul berkas di dalamnya.

Gambar 3.3 Hasil *Extract* Zip Bootstrap

6. *Framework* Bootstrap siap digunakan. Masukkan Bootstrap ini ke *folder project* dan mulailah membuat sebuah *project***.**

3.3 Sistem *Grid* pada Bootstrap

Sistem *grid* pada Bootstrap adalah suatu aturan baku yang digunakan untuk pembuatan dan penataan *layout* halaman dengan menggunakan rangkaian baris dan kolom untuk penempatan konten kita nantinya. Ada beberapa cara sistem kerja pada Bootstrap sebagai berikut:

1. Setiap kolom harus terletak di antara kode .container (*fixed-width*) atau .container-fluid (*full-width*).
2. Sistem *grid* pada Bootstrap memperbolehkan kita mengatur hingga 12 kolom dalam 1 baris, dengan catatan jumlah setiap kolom dalam 1 baris tersebut harus pas 12.
3. Sistem *grid* pada Bootstrap bersifat *responsive*, jadi kolom-kolomnya akan secara otomatis tersusun mengikuti ukuran layar.

Penggunaan kelas pada *grid* dapat digunakan pada layar yang lebih besar maupun yang lebih kecil dari batasan. Contohnya .col-md-* untuk ukuran layar 970px, akan tetapi bisa berfungsi dengan baik untuk layar yang lebih kecil (kalau kelas .col-xs-* atau .col-sm-* tidak digunakan) ataupun yang lebih besar (kalau kelas .col-lg-* tidak digunakan.

3.3.1 Kelas pada Grib

Tabel 3.1 Daftar Kelas Grib

Nama Kelas	Keterangan
Xs	Untuk Smartphone
Sm	Untuk Tablet
Md	Untuk Desktop
Lg	Untuk Desktop Besar

3.3.2 Struktur dasar *grid* Bootstrap

```
<div class=”row”>
<div class=”col-*-*”>isi kolom</div
</div>
```

Letakkan kode diatas, di antara **.container**.

```
<div class=”container”>
Letakkan struktur dasar grid disini
</div>
```

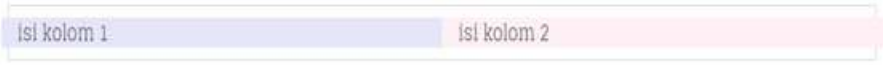
atau **.container-fluid**.

```
<div class=”container-fixed”
Letakkan struktur dasar grid disini
</div>
```

Tempatkan kode tersebut di antara **<!-- AWAL KONTEN -->**' sampai '**<!-- AKHIR KONTEN -->** Contohnya, bila ingin membuat 2 kolom, kodenya.

```
<div class=”row”>
<div class=”col-md-6”>isi kolom 1</div>
<div class=”col-md-6”>isi kolom 2</div>
</div>
```

Maka hasilnya jadi seperti ini:

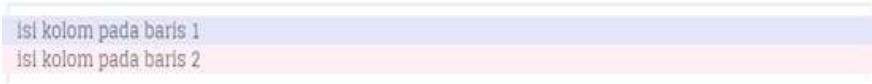


Gambar 3.4 *Output Container-fluid* 2 Kolom

Atau:

```
<div class=”row”>
<div class=”col-md-4”>isi kolom 1</div>
<div class=”col-md-8”>isi kolom 2</div>
</div>
```

Hasilnya jadi seperti ini.



Gambar 3.5 Output Container-fluid 2 Baris

3.3.3 Offsetting Kolom

Offsetting kolom bertujuan untuk memindahkan kolom ke kanan menggunakan kelas .col-md-offset-*. Kelas ini meningkatkan/menambah margin kiri suatu kolom sebanyak * kolom. Contohnya .col-md-offset-4 berarti memindahkan .col-md-4 sebanyak empat kolom. Contoh kodenya seperti ini.

```
<div class=”row”>
<div class=”col-md-4”>.col-md-4</div>
<div  class=”col-md-4  col-md-offset-4”>.col-md-4  .col-
md-offset-4</div>
</div>
<div class=”row”>
<div  class=”col-md-3  col-md-offset-3”>.col-md-3  .col-
md-offset-3</div>
<div  class=”col-md-3  col-md-offset-3”>.col-md-3  .col-
md-offset-3</div>
</div>
<div class=”row”>
<div  class=”col-md-6  col-md-offset-3”>.col-md-6  .col-
md-offset-3</div>
</div>
```

3.3.4 *Nesting* kolom (kolom bersarang)

Kolom bersarang di sini maksudnya kita menaruh kolom di dalam kolom. Untuk menaruh kolom di *grid* default, tambahkan .row baru dan set kolom .col="sm-* dengan kolom .col="sm-* yang sudah ada. Baris yang bersarang harus berisi kolom yang berjumlah 12 atau kurang (anda tidak

perlu menggunakan semua 12 kolom yang tersedia) Berikut ini kode yang dibutuhkan.

```
<div class=”row”>
<div class=”col-sm-9”>
Level 1: .col-sm-9
<div class=”row”>
<div class=”col-xs-8 col-sm-6”>
Level 2: .col-xs-4 .col-sm-6
</div>
</div>
</div>
</div>
```

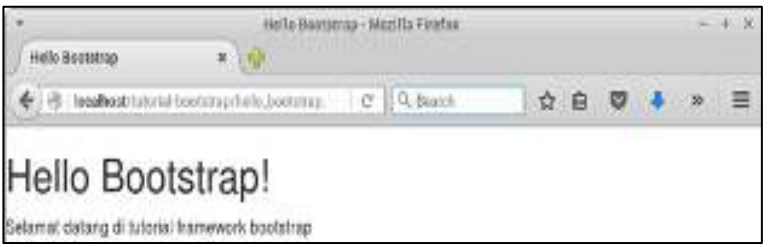
3.3.5 Penggunaan Sistem *Grid* pada Bootstrap

Sistem *grid* di Bootstrap mengadopsi konsep tabel. Karena itu, Kita hanya perlu menggunakan tiga kelas untuk membuatnya. Kelas tersebut kita terapkan dalam *tag* . Berikut ini tiga kelas yang dimaksud.

1. Kelas .container

Kelas kontainer berfungsi membungkus konten *web*. Kelas ini sama fungsinya seperti *tag* <table> dalam pembuatan tabel. Ada dua jenis kelas .container: (1) Kelas .container yang ukuran lebarnya tetap (*fixed*) dan (2) kelas .container-fluid yang ukuran lebarnya mengikuti lebar *browser*.

Pertama kita coba dulu kelas .container. Kita tinjau ulang hasil dari tutorial pertama. Konten *web* yang belum menggunakan kelas .container: hello_Bootstrap.html.



Gambar 3.6 Hasil Output Bootstrap

Selanjutnya, kita tambahkan *tag* dengan kelas .container dan kode CSS untuk *background* dan warna teks. Sehingga kodenya menjadi seperti berikut ini:

```
<!DOCTYPE html>
<html>
<head>
<title>Hello Bootstrap</title>
<!- menyisipkan Bootstrap ->
<link                                    rel=”stylesheet”
href=”Bootstrap/css/Bootstrap.min.css” />
</head>
<body>
<div   class=”container”   style=”background:   #008080;
color: white”>
<h1>Hello Bootstrap!</h1>
<p>Selamat datang di tutorial framework Bootstrap</p>
</div>
</body>
</html>
```

Maka, hasilnya akan seperti ini:

Gambar 3.7 *Output* Bootstrap dengan CSS

Konten *web* berpindah ke tengah, tidak lagi berada di tepi (samping). Itu disebabkan karena pengaruh dari kelas .container. Sekarang kita coba menggunakan kelas .container-fluid. Gantilah kelas *container* menjadi kelas *container-fluid*. Maka hasilnya akan seperti berikut ini.

Gambar 3.8 Output Kelas Container-fluid

Ukuran lebar kontainer akan mengikuti lebar *browser*. Coba saja perbesar ukuran *browser*, maka ukuran lebar kontainer akan ikut berubah (membesar). Satu hal lagi, di dalam sebuah *website* boleh memiliki lebih dari satu elemen kontainer. Jadi, bila anda ingin memisahkan kontainer untuk *header*, artikel, *footer*, dll. Itu dibolehkan.

2. Kelas .row

Di dalam kelas .container ada kelas lagi, yaitu kelas .row. Fungsinya untuk membuat baris. Bila dibandingkan dengan tabel, kelas ini seperti *tag* <tr>. Kita harus membuat elemen div dengan kelas *row* di dalam kontainer. Jangan membuatnya di luar. Contohnya sebagai berikut

```
<div class=”container”>
<div class=”row”>
… konten web disini …
</div>
</div>
```

3. Kelas .col

Di dalam elemen *row,* ada kelas .col-. Fungsinya untuk membuat kolom. Bila dibandingkan dengan tabel, kelas .col- seperti *tag* <td>. Kelas .col- memiliki ukuran-ukuran:

a. col-xs- (*extra small)* untuk perangkat dengan layar kecil seperti **ponsel**;

b. col-sm- (*small*) untuk perangkat dengan layar agak kecil seperti **tablet**;
c. col-md- (*medium*) untuk perangkat dengan layar sedang seperti **laptop**; dan
d. col-lg- (*large*) untuk perangkat dengan layar besar seperti **komputer (PC)**.

Jadi, agar ukurannya sesuai dengan perangkat yang digunakan, maka gunakanlah semuanya. Karena, sekarang *website* tidak hanya diakses melalui PC dan laptop saja. Namun, pada tutorial ini, kita sepakati menggunakan yang medium, yaitu col-md-.

Ada lagi ukuran yang harus diketahui, yaitu ukuran lebar kolom. Lebar kolom paling panjang adalah 12 dan paling pendek adalah 1. Untuk membuat kolom dengan lebar 12, kita cukup memanggil nama kelasnya .col-md-12. Pada dokumentasi Bootstrap, sudah dijelaskan seperti ini

Di dalam kelas .container ada kelas lagi, yaitu kelas .row. Fungsinya untuk membuat baris. Bila dibandingkan dengan tabel, kelas ini seperti *tag* <tr>. Kita harus membuat elemen div dengan kelas *row* di dalam kontainer. Jangan membuatnya di luar. Contohnya sebagai berikut:

4. Kelas .col-

Sekarang mari kita coba menerapkan konsep *grid* ini dengan membuat kode HTML untuk rancangan *web* seperti berikut ini

Header
Artikel
Footer

Pada rancangan di atas, terdapat tiga baris *(row)* dan satu kolom *(*col*)*.

3.4 Gambar *Preview* pada Bootstrap

Bootstrap menyediakan *class* untuk mengatur gambar sesuai dengan keinginan, misalnya membuat gambar dengan bentuk *rounder*, *circle*

(lingkaran) dan gambar yang *responsive*. Berikut ini dijelaskan tentang beberapa *class* Bootstrap yang bisa digunakan untuk membuat memanipulasi tampilan gambar pada Bootstrap.

1. **img-responsive**

   Class img-responsive berguna untuk membuat gambar menjadi responsif saat dijalankan pada semua bentuk dan ukuran resolusi.

2. **img-rounded**

   Class img-rounded digunakan untuk membuat gambar berbentuk round atau pada sisi sudut gambar memiliki bentuk melengkung.

3. **img-circle**

   Class img-circle digunakan untuk membuat gambar dengan bentuk lingkaran.

4. **img-thumbnail**

   Class img-thumbnail di gunakan untuk membuat gambar *thumbnail* dengan Bootstrap. Berikut ini beberapa contoh yang dapat dilihat untuk membuat tampilan gambar dengan Bootstrap.

   a. Membuat Gambar *Thumbnail* dengan Bootstrap

      Pada gambar di atas kita menggunakan *class* "img-thumbnail" untuk membuat gambar *thumbnail* dengan menggunakan Bootstrap.

Gambar 3.9 Tampilan Gambar dengan Bootstrap

b. Membuat Gambar *Circle* dengan Bootstrap
Sama dengan cara membuat gambar *thumbnail* pada contoh di atas. untuk membuat gambar dengan bentuk lingkaran anda dapat menggunakan *class* "img circle". tampilan gambar dengan Bootstrap.

Gambar 3.10 Tampilan Gambar *Circle* dengan Bootstrap

c. Membuat Gambar *Round* dengan Bootstrap

Gambar 3.11 Tampilan Gambar *Round* dengan Bootstrap

d. Membuat Gambar *Responsive* dengan Bootstrap
Untuk membuat gambar *responsive* dengan menggunakan Bootstrap anda dapat melakukannya dengan menambahkan *class* "img-responsive" pada elemen gambar. tampilan gambar dengan Bootstrap.

3.5 Soal Latihan

1. Installah Bootstrap dikomputer/laptop masing-masing!
2. Implementasikan *script* Bootstrap di dalam *form input* formulir pada hasil kerja bab 2 sebelumnya!

BAB IV

BOOTSTRAP DAN CSS

Capaian Pembelajaran

1. Mampu memahami konsep dari Bootstrap dan CSS.
2. Mampu mengimplementasikan penulisan kode Bootstrap.

4.1 Sistem *Grid* pada Bootstrap

Bootstrap mencakup sistem *grid fluida* pertama seluler yang responsif yang dapat meningkatkan hingga 12 kolom dengan tepat seiring meningkatnya ukuran perangkat atau *viewport*.

4.1.1 Sistem Kisi

Sistem kisi digunakan untuk membuat tata letak halaman melalui serangkaian baris dan kolom yang menampung konten Anda. Inilah cara kerja sistem kotak Bootstrap.

- Baris harus ditempatkan di dalam .container (lebar tetap) atau .container-fluid (penuh) untuk perataan dan bantalan yang tepat.
- Gunakan baris untuk membuat grup kolom horizontal.
- Konten harus ditempatkan di dalam kolom, dan hanya kolom yang bisa menjadi anak langsung dari baris.
- Kelas kisi yang telah ditentukan sebelumnya seperti .row dan .col-xs-4 tersedia untuk membuat tata letak kisi dengan cepat. Mixin yang lebih sedikit juga dapat digunakan untuk tata letak semantik yang lebih banyak.
- Kolom membuat talang (celah antara konten kolom) melalui *padding*. *Padding* itu diimbangi dalam baris untuk kolom pertama dan terakhir melalui margin negatif pada .row.

- Margin negatif adalah mengapa contoh di bawah ini tidak digunakan. Ini agar konten dalam kolom kisi dipagari dengan konten bukan kisi.
- Kolom kisi dibuat dengan menentukan jumlah dua belas kolom yang tersedia yang ingin anda span. Misalnya, tiga kolom yang sama akan menggunakan tiga .col-xs-4 .
- Jika lebih dari 12 kolom ditempatkan dalam satu baris, setiap kelompok kolom tambahan akan, sebagai satu unit, membungkus ke baris baru.
- Kelas kisi berlaku untuk perangkat dengan lebar layar lebih besar dari atau sama dengan ukuran *breakpoint*, dan menimpa kelas kisi yang ditargetkan pada perangkat yang lebih kecil. Oleh karena itu, mis. Menerapkan kelas .col-md-* pada suatu elemen tidak hanya akan mempengaruhi gaya pada perangkat menengah tetapi juga pada perangkat besar jika kelas .col-lg-* tidak ada.

4.1.2 Query Media

Query media digunakan untuk membuat breakpoint kunci dalam sistem *grid*.

```
/* Extra small devices (phones, less than 768px) */ /*
No media query since this is the
default in Bootstrap */ /* Small devices (tablets,
768px and up) */ @media ( min-width: @ screen-sm-min )
{ ... } /* Medium devices (desktops, 992px and up) */
@media ( min-width: @ screen-md-min ) { ... } /* Large
devices (large desktops, 1200px and up) */ @media (
min-width: @ screen-lg-min ) { ... }
```

4.1.3 Stacked-to-Horizontal

Menggunakan satu set kelas .col-md-* *grid*, Anda dapat membuat sistem *grid* dasar yang mulai ditumpuk pada perangkat seluler dan perangkat tablet (rentang ekstra kecil ke kecil) sebelum menjadi perangkat desktop (medium) horizontal. Tempatkan kolom kotak di sembarang .row.

```
<div class= "row" >
   <div class= "col-md-1" > .col-md-1 </div>
   <div class= "col-md-1" > .col-md-1 </div>
   <div class= "col-md-1" > .col-md-1 </div>
   <div class= "col-md-1" > .col-md-1 </div>
   <div class= "col-md-1" > .col-md-1 </div>
   <div class= "col-md-1" > .col-md-1 </div>
   <div class= "col-md-1" > .col-md-1 </div>
   <div class= "col-md-1" > .col-md-1 </div>
   <div class= "col-md-1" > .col-md-1 </div>
   <div class= "col-md-1" > .col-md-1 </div>
   <div class= "col-md-1" > .col-md-1 </div>
   <div class= "col-md-1" > .col-md-1 </div>
</div>
<div class= "row" >
   <div class= "col-md-8" > .col-md-8 </div>
   <div class= "col-md-4" > .col-md-4 </div>
</div>
<div class= "row" >
   <div class= "col-md-4" > .col-md-4 </div>
   <div class= "col-md-4" > .col-md-4 </div>
   <div class= "col-md-4" > .col-md-4 </div>
</div>
<div class= "row" >
   <div class= "col-md-6" > .col-md-6 </div>
   <div class= "col-md-6" > .col-md-6 </div>
</div>
<div class= "row" >
   <div class= "col-md-1" > .col-md-1 </div>
   <div class= "col-md-1" > .col-md-1 </div>
   <div class= "col-md-1" > .col-md-1 </div>
   <div class= "col-md-1" > .col-md-1 </div> `
   <div class= "col-md-1" > .col-md-1 </div>
   <div class= "col-md-1" > .col-md-1 </div>
   <div class= "col-md-1" > .col-md-1 </div>
   <div class= "col-md-1" > .col-md-1 </div>
   <div class= "col-md-1" > .col-md-1 </div>
   <div class= "col-md-1" > .col-md-1 </div>
   <div class= "col-md-1" > .col-md-1 </div>
   <div class= "col-md-1" > .col-md-1 </div>
</div>
<div class= "row" >
```

```
   <div class= "col-md-8" > .col-md-8 </div>
   <div class= "col-md-4" > .col-md-4 </div>
</div>
<div class= "row">
   <div class= "col-md-4" > .col-md-4 </div>
   <div class= "col-md-4" > .col-md-4 </div>
   <div class= "col-md-4" > .col-md-4 </div>
</div>
<div class= "row" >
   <div class= "col-md-6" > .col-md-6 </div>
   <div class= "col-md-6" > .col-md-6 </div>
</div>
```

4.1.4 Wadah Cairan

Mengubah tata letak kisi lebar tetap menjadi tata letak lebar penuh dengan mengubah .container terluar Anda menjadi .container-fluid.

```
<div class= "container-fluid" >
<div class= "row" > ... </div> </div>
```

4.1.5 Kolom Bersarang

```
<div class= "row" >
<div class= "col-sm-9" > Level 1: .col-sm-9
div class= "row" >
div class= "col-xs-8 col-sm-6" >
Level 2: .col-xs-8 .col-sm-6
</div>
<div class="col-xs-4 col-sm-6" >
Level 2: .col-xs-4 .col-sm-6
</div>
</div>
</div>
div>
```

4.2 Tipografi

4.2.1 Pos

Semua judul HTML, <h1> hingga <h6>, tersedia. Kelas h1 hingga .h6 juga tersedia, untuk saat Anda ingin mencocokkan gaya *font* judul tetapi tetap ingin teks Anda ditampilkan secara *inline*.

4.2.2 Salinan Tubuh

Font-size default global Bootstrap adalah **14px**, dengan *line-height* **1,428**. Ini diterapkan pada <body> dan semua paragraf. Selain itu, <p> (paragraf) menerima margin bawah setengah tinggi garis (secara *default* 10px).

4.2.3 Elemen Teks

a. Teks yang Ditandai
   Untuk menyorot serangkaian teks karena relevansi dalam konteks lain, gunakan *tag* <mark>.
b. Teks yang Dihapus
   Untuk menunjukkan blok teks yang telah dihapus gunakan *tag* <del>.
c. Teks yang Dicoret
   Untuk menunjukkan blok teks yang telah dihapus gunakan *tag* <del>.
d. Teks yang Digarisbawahi
   Untuk menggaris bawahi teks, gunakan *tag* <u>.

4.3 Soal Latihan

1. Buatlah ringkasan tentang Bootstrap ke dalam Google Docs, lengkap dengan contoh pembuatannya, dengan menggunakan bahasa sendiri!

BAB V

PHP

Capaian Pembelajaran

1. Mampu menggunakan perangkat lunak yang dibutuhkan untuk membuat halaman *web* dengan PHP.
2. Mampu membuat halaman *web* sederhana dengan PHP.
3. Mengenal tipe data, variabel.
4. Menggunakan operator.

5.1 Pengertian PHP

PHP atau merupakan singkatan rekursif dari PHP: Hyper*text Preprocessor* adalah suatu bahasa pemrograman yang termasuk kategori *server-side programming* (Muhardin, 2003). *Server-side programming* adalah jenis bahasa pemrograman yang nantinya *script/program* tersebut akan dijalankan oleh server. Selanjutnya hasil pengolahan *script/program* tersebut akan dikirim ke *client* sebagai *output*.

Selain *server-side programming*, PHP juga memiliki *client-side programming*. Jenis *programming* ini merupakan kebalikan dari *server-side programming*. Untuk *client-side programming*, *script/program* akan diproses di dalam *client* sendiri.

5.2 Kebutuhan Perangkat Lunak

Untuk membuat *web* dengan PHP dibutuhkan perangkat lunak sebagai berikut.

1. *Editor*, yaitu digunakan untuk menuliskan **script** PHP. Contoh: Notepad, Textpad, Notepad++, Dreamweaver, dll

2. *Web server* yaitu digunakan untuk memberikan layanan *web*, memproses dan menjalankan perintah *script* PHP. Contoh: Apache, IIS, Nginx, dll.
3. PHP yaitu bahasa skrip yang di jalankan di *web server* untuk menerjemahkan dokumen PHP dan berinteraksi dengan basis data melalui API.
4. MySQL yaitu *database* manajemen sistem yang digunakan untuk mengelola *database* aplikasi yang dibuat.
5. *Web browser* yaitu aplikasi di pengguna untuk menjalankan dan menampilkan halaman *web*. Contoh: Mozilla Firefox, Internet Explorer, Google Chrome, Safari dsb.

Untuk memudahkan instalasi perangkat lunak yang digunakan dalam membuat *web* dengan PHP, saat ini sudah tersedia perangkat lunak paket yang terdiri atas *web* server, PHP dan MySQL serta perangkat lunak pendukung lainnya. Contoh: XAMPP, Appserv, Wampp dan sebagainya.

5.3 Membuat Halaman *Web* Sederhana dengan PHP

Pastikan *web server* dan skrip PHP anda telah berjalan dengan baik sebelum memulai menuliskan *script* PHP. Untuk membuat *web* dengan *script* PHP, cukup mempersiapkan *editor* teks. Fungsi-fungsi yang ada di PHP bersifat *uncase sensitive*, tetapi variabelnya *case sensitive* (membedakan huruf besar dan kecil).

Terdapat beberapa cara untuk menulis *script* PHP. Terdapat *tag* pembuka dan penutup yang menyatakan PHP untuk memulai atau mengakhiri apa yang akan diinterpretasikan melalui *web browser*. Beberapa contohnya akan diberikan di bawah ini.

5.4 Jenis-Jenis *Tag* PHP

PHP adalah bahasa *tag*, di mana setiap *file* yang isinya menggunakan bahasa PHP pasti di awali dan di akhiri oleh *tag* PHP, ada dua jenis *tag* dalam bahasa pemrograman PHP yaitu *tag* pembuka **<?php** dan *tag* penutup?**>** . penulisan *tag* ini sangat dibutuhkan karena *interpreter*

PHP dalam bekerja akan mencari *tag* PHP, *interpreter* PHP akan memulai menerjemahkan kode awal PHP dari *Tag* pembuka PHP dan mengakhiri penerjemahannya pada *tag* penutup PHP.

Tabel 5.1 *Tag* PHP

***Tag* pembuka**	***Tag* penutup**
<?php	?>
<?	?>
<script language=”php”>	</script>

Contoh penggunaan adalah sebagai berikut.

1. Pada contoh di bawah ini, tulisan yang akan di-*parsing* oleh PHP adalah “*server-side scripting*”. Sedangkan tulisan “Mari kita belajar” dan “menggunakan PHP” merupakan tulisan yang ditulis menggunakan HTML. Dalam contoh ini, PHP disisipkan pada *tag* HTML.

```
<p>Mari kita belajar
<?php
    echo "server-side scripting ";
?>
    menggunakan PHP </p>
```

2. Penulisan lain yang sedikit lebih variatif dijabarkan di bawah ini. PHP hanya akan menampilkan salah satu dari kalimat “Rajin pangkal pandai.” atau “Hemat pangkal kaya.” tergantung nilai *True* / *false* dari variabel $pilih.

```
<?php
if ($pilih) {
?>
 <i> Rajin pangkal pandai. </i>
 <?php }
else{
?>
<b> Hemat pangkal kaya. </b>
```

```
<?php
}?>
```

5.5 Penggunaan Komentar pada PHP

PHP memberikan cara agar *programer* dapat membuat komentar pada skripnya. Komentar ini pun dapat dibuat untuk yang hanya 1 baris *script* (*one-line comment*) maupun beberapa baris *script* (*multi-line comment*).

Tabel 5.2 Kode Komentar pada PHP

Tanda	Keterangan
//	echo "komentar PHP"; // contoh dengan jenis one-*line* c style
#	echo "komentar PHP"; # contoh dengan jenis one-*line* shell style
/* */	echo "komentar multi *line*"; /* ini merupakan contoh penggunaan multi-*line* comment yang memberikan komentar untuk beberapa baris kode PHP */

5.6 Penulisan Karakter Khusus dengan tanda

Karakter yang ditulis dengan diawali tanda (\) yaitu sebagai berikut.

Tabel 5.3 Karakter Khusus PHP

Karakter	Keterangan	Karakter	Keterangan
\”	Tanda petik ganda	\\	Tanda *backslash*
\$	Tanda $	\n	Pindah baris
\t	tab	\x00 s.d \xFF	heksadesimal

5.7 Variabel PHP

Variabel pada PHP direpresentasikan oleh karakter dolar ($) dan dilanjutkan dengan nama *variable* tersebut. Variabel pada PHP bersifat *case-sensitive*. Nama *variable* yang valid pada php dimulai dengan huruf atau *underscore* (_) dan diikuti oleh huruf, angka atau *underscore*. Jika dilihat sebagai *regular expression*: [a-zA-Z_\x7f-\xff][a-zA-Z0-9_\x7f-\xff]*

Huruf terdiri dari a sampai z dan A sampai Z, karakter ASCII dari 127 sampai 255 (0x7f sampai 0xff). *Variable* tidak boleh menggunakan tanda baca ataupun *reserved word* PHP, seperti *print*, *echo*.

5.8 Operator

Dalam PHP juga dapat melakukan proses operasi, baik itu penjumlahan, operasi logika, ataupun operasi pembanding. Operator Matematika yang digunakan dalam PHP yaitu sebagai berikut.

Tabel 5.4 Operator PHP

Operator	Fungsi	Operator	Fungsi
+	Penjumlahan	-	Pengurangan
*	Perkalian	/	Pembagian
%	Sisa pembagian	++ , --	Penaikan, penurunan

Operator Pembanding yang digunakan dalam PHP yaitu sebagai berikut.

Tabel 5.5 Operator Pembanding

Operator	Fungsi	Operator	Fungsi
==	Sama dengan	<	Kurang dari
>	Lebih dari	<=	Kurang dari atau sama dengan
>=	Lebih dari atau sama dengan	!= , <>	Tidak sama dengan

Selain itu, operator Logika juga dapat digunakan di PHP, antara lain, and (&&), or (||), xor, dan!

5.9 Soal Latihan

1. Buatlah ringkasan tentang PHP ke dalam Google Docs, lengkap dengan contoh penggunaannya, dengan menggunakan bahasa sendiri!

BAB VI

DATABASE

Capaian Pembelajaran

1. Mampu memahami apa itu *database*.
2. Mampu memahami tipe data.
3. Mampu membuat *database*.
4. Mampu membuat tabel pada *database*.
5. Mampu membuat *web* yang terhubung ke *database*.

6.1 Pengertian *Database*

Database atau sistem basis data adalah suatu sistem menyusun dan mengelola *record-record* menggunakan komputer untuk menyimpan atau merekam serta memelihara data (Marlinda, 2004). Keuntungan sistem basis data adalah:

1. mengurangi kerangkapan data,
2. mencegah ketidakkonsistenan,
3. integritas dapat dipertahankan, dan
4. data dapat digunakan bersama-sama.

6.2 Database MySQL

MySQL merupakan sebuah perangkat lunak/*software* sistem manajemen basis data SQL atau DBMS Multithread dan *multi user*. MySQL sebenarnya merupakan turunan dari salah satu konsep utama dalam *database* untuk pemilihan atau seleksi dan pemasukan data yang memungkinkan pengoperasian data dikerjakan secara mudah dan otomatis. MySQL diciptakan oleh Michael "Monty" Widenius pada tahun 1979, seorang *programmer* komputer asal Swedia yang mengembangkan sebuah

sistem *database* sederhana yang dinamakan UNIREG yang menggunakan koneksi *low-level* ISAM *database engine* dengan *indexing*.

6.3 Jenis-Jenis Tipe Data *Database* MySQL

Secara garis besar, *database* MySQL mempunyai 3 macam tipe data, yaitu tipe data *numeric*, tipe data *date* & *time*, dan tipe data *string*.

6.3.1 Tipe Data *Numeric*

Tipe Data *Numeric* pada *database* MySQL terbagi atas beberapa macam tipe data, yaitu sebagai berikut.

1. INT
   Digunakan untuk menyimpan data yang berupa bilangan bulat positif dan negatif dengan jangkauan antara -2.147.483.648 s.d. 2.147.483.647. Tipe data ini mempunyai ukuran 4 *byte* (32 bit). Contoh: TOTAL_MAHASISWA INT;
2. TINYINT
   Digunakan untuk menyimpan data yang berupa bilangan bulat **positif** dan negatif dengan jangkauan antara -128 s.d. 127. Tipe data ini mempunyai ukuran 1 *byte* (8 bit).
3. SMALLINT
   Digunakan untuk menyimpan data yang berupa bilangan bulat positif dan negatif dengan jangkauan antara -32.768 s.d. 32.767. Tipe data ini mempunyai ukuran 2 *byte* (16 bit) .
4. MEDIUMINT
   Digunakan untuk menyimpan data yang berupa bilangan bulat positif dan negatif dengan jangkauan antara -8.388.608 s.d. 8.388.607. Tipe data ini mempunyai ukuran 3 *byte* (24 bit).
5. BIGINT
   Digunakan untuk menyimpan data yang berupa bilangan bulat positif dan negatif dengan jangkauan antara -8.388.608 s.d. 8.388.607. Tipe data ini mempunyai ukuran 8 byte (64 bit).

6. FLOAT
   Digunakan untuk menyimpan data yang berupa bilangan pecahan positif dan negatif presisi tunggal. Tipe data ini mempunyai ukuran 4 *byte* (32 bit).
7. DOUBLE
   Digunakan untuk menyimpan data yang berupa bilangan pecahan positif dan negatif presisi ganda. Tipe data ini mempunyai ukuran 8 *byte* (64 bit).
8. DECIMAL
   Digunakan untuk menyimpan data yang berupa bilangan pecahan positif dan negatif presisi ganda. Tipe data ini mempunyai ukuran 8 *byte* (64 bit).
9. REAL
   Digunakan untuk menyimpan data yang berupa bilangan pecahan positif dan negatif. Tipe data ini mempunyai ukuran 8 *byte* (64 bit).
10. NUMERIC
    Digunakan untuk menyimpan data yang berupa bilangan pecahan positif dan negatif. Tipe data ini mempunyai ukuran 8 *byte* (64 bit).

6.3.2 Tipe Data *Date* & *Time*

Tipe Data *Date & Time* pada *database* MySQL terbagi atas beberapa macam tipe data, yaitu sebagai berikut.

1. DATE
   Digunakan untuk meyimpan data tanggal dalam format YY:MM:DD.
2. DATETIME
   Digunakan untuk menyimpan data tanggal dan waktu dalam format YY:MM:DD HH:MM:SS.
3. TIME
   Digunakan untuk menyimpan data waktu dalam format HH:MM:SS.
4. YEAR
   Digunakan untuk menyimpan data tahun.

6.3.3 Tipe Data String

Tipe Data *String* pada *database* MySQL terbagi atas beberapa macam tipe data, yaitu sebagai berikut.

1. CHAR
   Digunakan untuk menyimpan data karakter/*string* dengan ukuran tetap. Tipe data ini mempunyai jangkauan antara 0 sampai dengan 255 karakter.
2. VARCHAR
   Digunakan untuk menyimpan data karakter/*string* dengan ukuran dinamis. Tipe data ini mempunyai jangkauan antara 0 sampai dengan 255 untuk MySQL versi 4.1. Dan mempunyai jangkauan antara 0 s.d. 65.535 untuk MySQL versi 5.0.3.
3. BLOB
   BLOB (*Binary Large Object*) adalah tipe data yang digunakan untuk menyimpan data biner seperti *file*s, *images*, suara, dll. Tipe data ini mempunyai jangkauan 216-1 byte.
4. TINYBLOB
   Digunakan untuk menyimpan data biner seperti *file*, *image*, dan suara. Tipe data ini mempunyai jangkauan 255 *byte*.
5. MEDIUMBLOB
   Digunakan untuk menyimpan data biner seperti *file*, *image*, dan suara. Tipe data ini mempunyai jangkauan 244-1 *byte*.
6. LONGBLOB
   Digunakan untuk menyimpan data biner seperti *file*, *image*, dan suara. Tipe data ini mempunyai jangkauan 232-1 *byte*.
7. TEXT
   Digunakan untuk menyimpan data *text*. Tipe data ini mempunyai jangkauan antara 0 sampai dengan 65.535 (216-1) karakter.
8. TINYTEXT
   Digunakan untuk menyimpan data *text*. Tipe data ini mempunyai jangkauan antara 0 s.d. 255 untuk MySQL versi 4.0, dan mempunyai jangkauan antara 0 s.d. 65.535 untuk MySQL versi 5.0.3.

9. MEDIUMTEXT
   Digunakan untuk menyimpan data *text*. Tipe data ini mempunyai jangkauan antara 0 sampai dengan 224-1 karakter.
10. LONGTEXT
    Digunakan untuk menyimpan data *text*. Tipe data ini mempunyai jangkauan antara 0 sampai dengan 232-1 karakter.
11. ENUM
    Digunakan untuk menyimpan data enumerasi (kumpulan data).
12. SET
    Digunakan untuk menyimpan data himpunan data.

6.4 Langkah Pembuatan *Database* MySQL

Hal–hal yang perlu di perhatikan saat membuat *database* adalah sebagai berikut.

1. Mengumpulkan semua informasi yang dibutuhkan untuk membuat *Website*.
2. Melakukan survei ke *website* yang sudah terkenal yang 1 tema dengan *website* yang akan dibuat. Contoh *website* terkenal dalam hal penjualan adalah http://www.tokobagus.com.
3. Menyimpulkan hasil informasi dan survei.
4. Merancang *database* dan membuat strukturnya seperti Tabel, *Field*, pengambilan *type* data dan lain-lain.

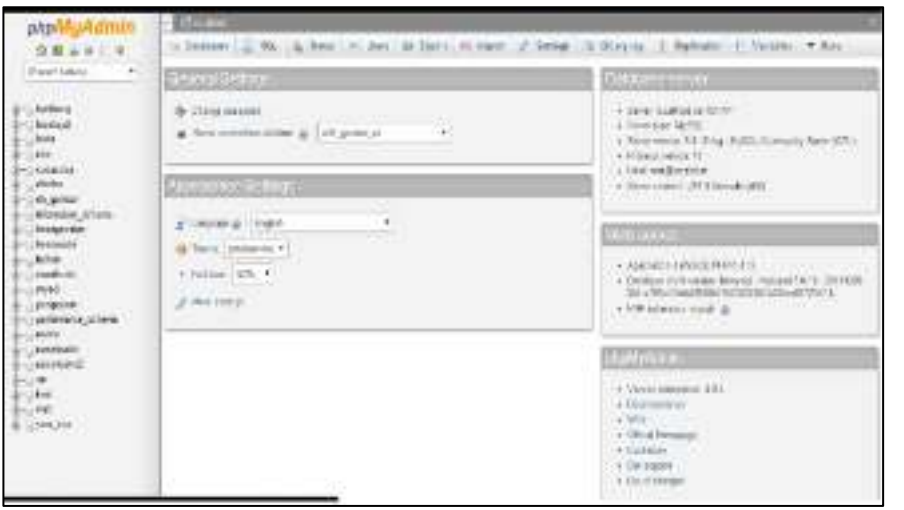
6.4.1 Cara Membuka Phpmyadmdin

1. Nyalakan *service Wampserver* dengan mengeklik 2x *file* "Wampmanager.exe" pada direktori C:\wamp.
2. Buka *Web Browser* (Mozilla Firefox, Google Chrome, ataupun Operamini).
3. Pada *address bar* tuliskan "localhost/phpmyadmin" atau "127.0.0.1/ phpmyadmin ".

4. Tampilan *login* di phpmyadmin ada pada Gambar 6.1.

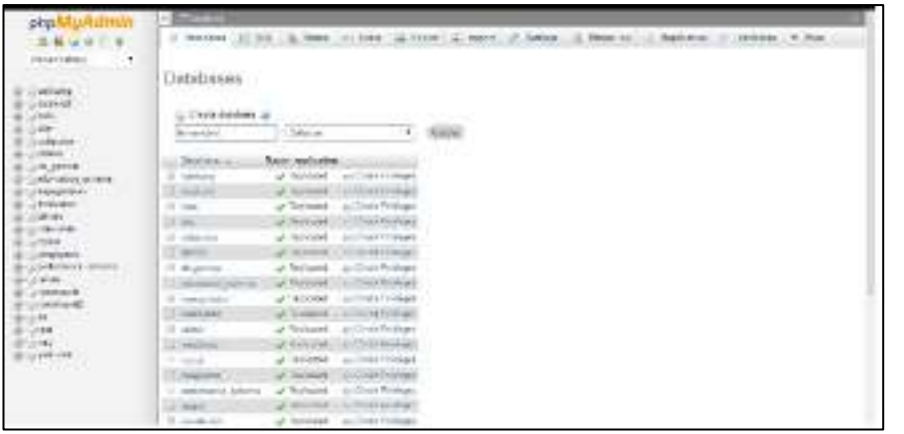
Gambar 6.1 *Login* phpmyadmin

5. Halaman setelah *login* phpmyadmin.



Gambar 6.2 Setelah *Login*

6. Membuat *database* dengan nama "fnmnewbie".



Gambar 6.3 Membuat *Database*

7. Membuat Tabel dengan nama fnmnewbie_tabel dan banyak *field* 3.

Gambar 6.4 Membuat Tabel

8. Membuat 3 *field* dengan rincian sebagai berikut.
   a. No bertipe INT.
   b. Nama bertipe VARCHAR sepanjang 30 huruf.
   c. Kelas bertipe VARCHAR sepanjang 5 huruf.

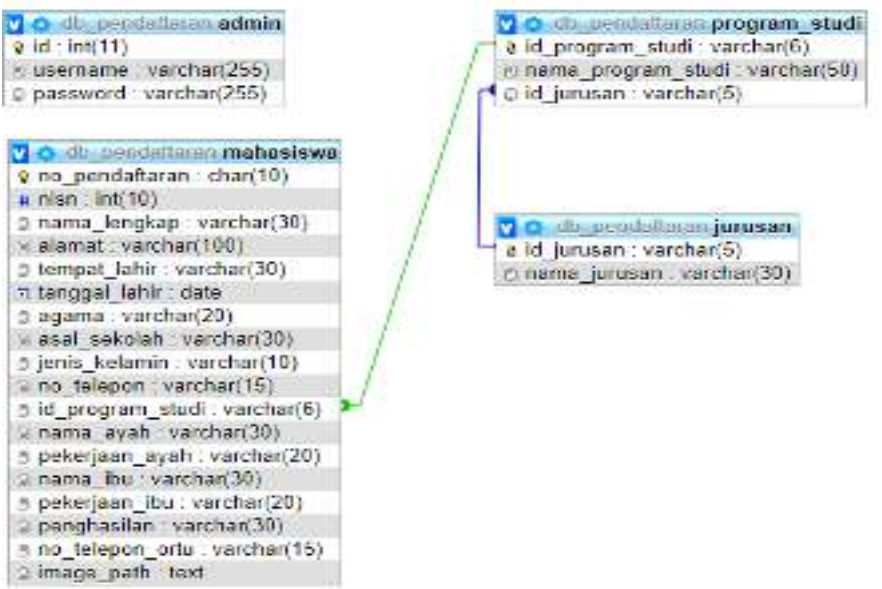
Gambar 6.5 Membuat *Field* pada Tabel

9. Tampilan setelah berhasil membuat *table*.

Gambar 6.6 Data Pada Tabel

6.5 Penyederhanaan Struktur *Database*

Berikut contoh tabel basis data.



Gambar 6.7 Relasi *Database*

Untuk menghilangkan kerangkapan data, untuk mengurangi kompleksitas dan untuk mempermudah pemodifikasian data. Data diuraikan dalam bentuk tabel, selanjutnya dianalisis berdasarkan persyaratan tertentu ke beberapa tingkat dan apabila tabel yang diuji tidak memenuhi persyaratan tertentu, maka tabel tersebut perlu dipecah menjadi beberapa tabel yang lebih sederhana sampai memenuhi bentuk yang optimal. Untuk lebih jelas dapat dilihat pada Gambar 6.8 di bawah ini yang membagi tabel menjadi tiga bagian.

Gambar 6.8 Daftar Tabel

6.6 Menentukan *Has One*

6.6.1 Konsep *Has One*

Hubungan satu-ke-satu diimplementasikan menggunakan *has one* metode ini. Misalnya, kita punya *user* model. Setiap pengguna memiliki satu *profile*, dan *user* tabel harus dikaitkan dengan *profile* tabel. Untuk dapat menemukan profil untuk pengguna tertentu, kita harus menambahkan metode yang dipanggil *profile* ke *user* kelas (perhatikan bahwa nama metode di sini sewenang-wenang, tetapi harus menggambarkan hubungannya). Metode ini memanggil metode yang dilindungi *has one* yang disediakan oleh Paris, dengan meneruskan nama kelas objek terkait. *The profile* metode harus mengembalikan *instance* ORM siap (opsional) penyaringan lebih lanjut.

```
<?
Profil kelas php meluas Model { }
kelas Pengguna memperluas Model {
profil fungsi publik () { return $ this -> has_one
( 'Profil' );
    }
}
```

API untuk metode ini berfungsi sebagai berikut.

```
<? php
    // Pilih pengguna tertentu dari basis data
$ user = Model:: factory ( 'User' ) -> find_one ( $
user_id );
    // Cari profil yang dikaitkan dengan pengguna
$ profile = $ user -> profile () -> find_one ();
```

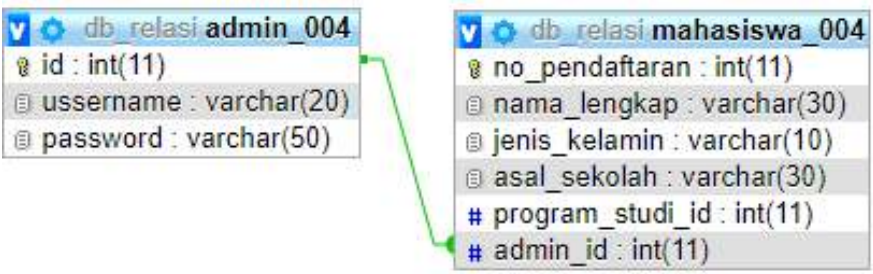
Secara *default*, Paris mengasumsikan bahwa kolom kunci asing pada tabel terkait memiliki nama yang sama dengan tabel (pangkalan) saat ini, dengan _id menambahkan. Pada contoh diatas, Paris akan mencari kolom kunci asing yang dipanggil *user id* pada tabel yang digunakan oleh *profile* kelas. Untuk mengganti perilaku ini, tambahkan argumen kedua ke *has one* panggilan Anda dengan meneruskan nama kolom yang akan digunakan.

Selain itu, Paris mengasumsikan bahwa kolom kunci asing di saat ini (pangkalan) tabel adalah kolom kunci utama dari tabel dasar. Dalam contoh di atas,

Paris akan menggunakan kolom yang disebut *user id* (dengan asumsi *user id* adalah kunci utama untuk tabel pengguna) di tabel dasar (dalam hal ini tabel pengguna) sebagai kolom kunci asing di tabel dasar. Untuk mengganti perilaku ini, tambahkan argumen ketiga, melewati nama kolom yang ingin Anda gunakan sebagai kolom kunci asing di tabel dasar *has one call*.

6.6.2 Relasi Antartabel

Data relasional adalah suatu model data yang meletakkan data dalam bentuk relasi (atau popular dengan sebutan *table*). Di dalam model data relasional dikenal istilah relasi (*relation*). Yang disebut relasi adalah tabel yang terdiri atas baris dan kolom.



Gambar 6.9 Relasi Antar Tabel

6.7 *Query* Paris ORM

ORM (*Object Relational Mapping*) adalah metode pemrograman yang digunakan untuk mengonversi data dari lingkungan bahasa pemrograman berorientasi objek (OOP) dengan lingkungan *database* relasional. berikut contoh penulisan kode *query* ParisORM.

```
<?php
require_once ('config.php');
Class adminORM extends Model
    {
    public static $_table ='admin_004';
    public function mahasiswa()
    {
    return $this->has_one('mahasiswaORM');
}
```

6.8 Menentukan *Has Many*

Hubungan satu-ke-banyak diimplementasikan menggunakan *has many* metode ini. Misalnya, kita punya *user* model. Setiap pengguna memiliki beberapa *Post* objek. The *user table* harus dikaitkan dengan *post* meja. Untuk dapat menemukan *posting* untuk pengguna tertentu, kita harus menambahkan metode yang dipanggil *posts* ke *User* kelas (perhatikan bahwa nama metode di sini sewenang-wenang, tetapi harus menggambarkan hubungannya). Metode ini memanggil metode terlindungi *has many* yang disediakan oleh Paris, dengan meneruskan nama kelas

objek terkait. Lulus nama kelas model secara harfiah, bukan versi jamak. *The posts* metode harus mengembalikan *instance* ORM siap (opsional) penyaringan lebih lanjut.

```
<? php
class Post extends Model {
}
kelas Pengguna memperluas Model {
 posting fungsi publik () { return $ this -> has_many (
'Posting' ); // Catatan kami menggunakan nama model
secara harfiah - bukan versi jamak
}
}
```

API untuk metode ini berfungsi sebagai berikut.

```
<? php
// Pilih pengguna tertentu dari basis data
$ user = Model:: factory ( 'User' ) -> find_one ( $
user_id );
// Temukan tulisan yang dikaitkan dengan pengguna
$ posts = $ user -> posts () -> find_many ();
```

Secara *default*, Paris mengasumsikan bahwa kolom kunci asing pada tabel terkait memiliki nama yang sama dengan tabel (pangkalan) saat ini, dengan _id menambahkan. Pada contoh di atas, Paris akan mencari kolom kunci asing yang dipanggil user_id pada tabel yang digunakan oleh *Post* kelas. Untuk mengganti perilaku ini, tambahkan argumen kedua ke has_many dengan meneruskan nama kolom yang akan digunakan.

Selain itu, Paris mengasumsikan bahwa kolom kunci asing dalam tabel (basis) saat ini adalah kolom kunci utama dari tabel basis. Dalam contoh di atas, Paris akan menggunakan kolom yang disebut user_id (dengan asumsi user_id adalah kunci utama untuk tabel pengguna) di tabel dasar (dalam hal ini tabel pengguna) sebagai kolom kunci asing di tabel dasar. Untuk mengganti perilaku ini, tambahkan argumen ketiga ke Anda,

melewati nama kolom yang ingin Anda gunakan sebagai kolom kunci asing di tabel dasar has_many *call*.

6.9 Menentukan *Belongs to*

'Sisi lain' dari has_one dan has_many adalah *belongs to*. Panggilan metode ini mengambil parameter identik sebagai metode ini, tetapi mengasumsikan kunci asing adalah pada *saat* (*base*) tabel, bukan tabel terkait.

```
<? php
class Profile extends Model {
pengguna fungsi publik () { return $ this -> milik_to (
'Pengguna' ); } }
kelas Pengguna memperluas Model { }
```

API untuk metode ini berfungsi sebagai berikut.

```
<? php
// Pilih profil tertentu dari database
$ profile = Model:: factory ( 'Profile' ) -> find_one (
$ profile_id );
// Cari pengguna yang terkait dengan profil
$ user = $ profile -> user () -> find_one ();
<? php
// Pilih profil tertentu dari database
$ profile = Model:: factory ( 'Profile' ) -> find_one (
$ profile_id );
// Cari pengguna yang terkait dengan profil
$ user = $ profile -> user () -> find_one ();
```

Sekali lagi, Paris membuat asumsi bahwa kunci asing pada tabel (pangkalan) saat ini memiliki nama yang sama dengan tabel terkait dengan _id ditambahkan. Pada contoh di atas, Paris akan mencari kolom bernama user_id. Untuk mengganti perilaku ini, berikan argumen kedua ke *belongs to* metode, tentukan nama kolom pada tabel (dasar) saat ini untuk digunakan.

Paris juga membuat asumsi bahwa kunci asing dalam tabel terkait (terkait) adalah kolom kunci utama dari tabel terkait. Pada contoh di atas, Paris akan mencari kolom bernama user_id di tabel pengguna (tabel terkait dalam contoh ini). Untuk mengganti perilaku ini, berikan argumen ketiga ke metode hers_to, tentukan nama kolom dalam tabel terkait untuk digunakan sebagai kolom kunci asing di tabel terkait.

6.10 Soal Latihan

1. Buatlah 1 buah *form input* dan *list* data inputnya, berupa tabel yang terhubung ke dalam *database*, dengan menggabungkan *script* html, css dan Bootstrap pada *form input* dan *list* tersebut!
2. Buatlah ringkasan tentang *database* mysql ke dalam Google Docs, lengkap dengan contoh pembuatannya, dengan menggunakan bahasa sendiri!

BAB VII

PARIS ORM COMPOSER

Capaian Pembelajaran

1. Mampu memahami paris ORM.
2. Mampu memahami kegunaan paris ORM.
3. Mampu menginstal paris ORM.
4. Mampu menggunakan *library carbon*.

7.1 Paris ORM *Composer*

ParisORM merupakan *library* aksesibilitas dan manipulasi *database* agnostik (dalam artian tidak terikat pada satu *framework* tertentu) yang jauh lebih mudah daripada menggunakan *active record* bawaan dari Codeigniter. Secara sederhana jika menggunakan ParisORM sama seperti menggunakan *library* EloquentORM dari Laravel, sehingga dengan menggunakan ParisORM tidak perlu pakai Laravel.

7.2 Cara Install Paris ORM Composer

1. Instal aplikasi *composer* silakan *download composer setup* terbaru pada https://getcomposer.org/.

Gambar 7.1 Download Composer

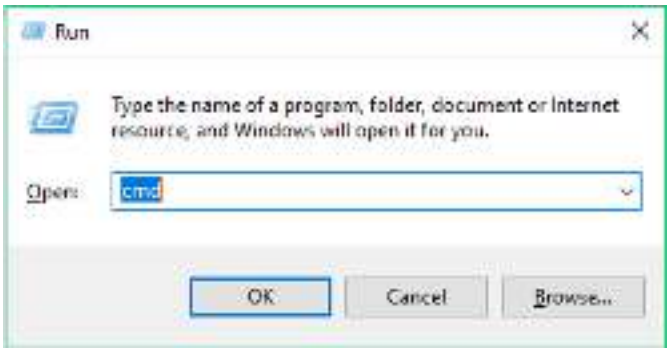
2. Dalam proses instalasi cukup klik *next* hingga muncul tampilan pemilihan penyimpanan *file*.
3. Pilih *folder directory* penyimpanan *file* php.exe, dicontoh ini *file* disimpan di *directory* C: maka *file* php.exe akan berada pada C:xampp\php\php.exe.

Gambar 7.2 Directory Penyimpanan *File*

4. Setelah kita *install*, buka *command prompt* dan ketik "*composer*". Jika berhasil tampilannya akan seperti ini.

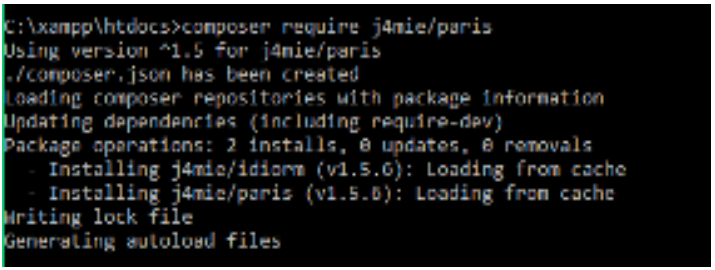
Gambar 7.3 Hasil *Install Composer*

5. Cara menginstal *Vendor* untuk *autoload* ParisORM. Langkah pertama masuk *command prompt* pada windows dengan menekan Ctrl+R dan tuliskan cmd lalu klik OK.



Gambar 7.4 Jendela *Command Prompt*

6. Setelah masuk *command prompt*, maka kita langsung diarahkan ke posisi *directory default*.
7. Selanjutnya keluar dari *folder* diatas hingga sampai ke *folder* C:\> dengan mengetikkan perintah 'cd..' (tanpa koma).
8. Selanjutnya masuk ke *folder* yang ingin di berikan instalasi ParisORM, dan untuk mengecek *list folder* yang ada di *directory* saat ini maka tuliskan perintah 'dir', yang berfungsi untuk menampilkan *folder* apa saja yang ada pada *directory* tersebut, seperti gambar di bawah ini.
9. Selanjutnya masuk ke *folder* yang diinginkan, dengan mengetikkan perintah 'cd".
10. Setelah memilih *folder*, selanjutnya adalah instal ParisORM dengan cara mengetikkan '*composer require* j4mie/paris' lalu klik *enter*. Penulisan *require* tidak selalu sama, tergantung jenis ParisORM yang ingin dipakai, untuk melihat cara penulisan *require* sesuai paris yang diinginkan bisa dilihat di *website* https://packagist.org/. Jika penginstalan sudah selesai maka akan tampil tulisan seperti gambar di bawah ini.

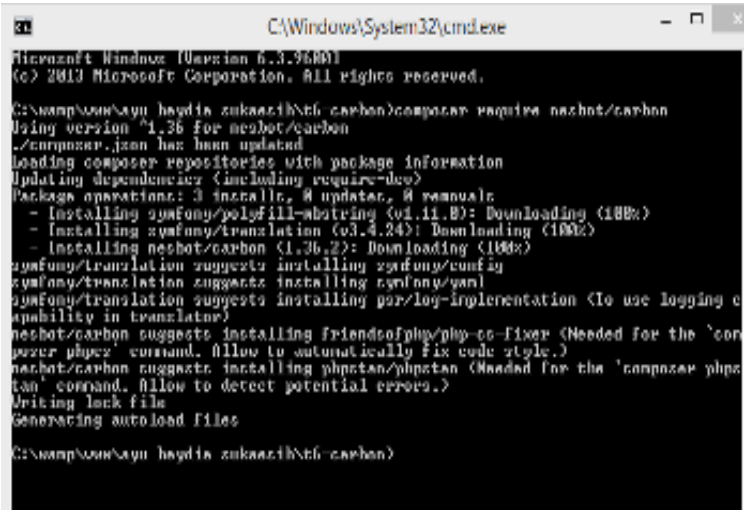


Gambar 7.5 *Output Install Composer* Selesai

11. Jika sudah selesai maka kita tinggal mengaplikasikan penggunaan ParisORM pada *coding* php yang akan kita buat. Akan tetapi apabila anda menemukan *error* pada saat penginstalan ParisORM maka anda perlu mengubah xampp ke wamp server versi 3.1.7 32 bit.

7.3 Membuat Format Tanggal dari Hari, Bulan, Tahun Dengan Menggunakan *Library Carbon*

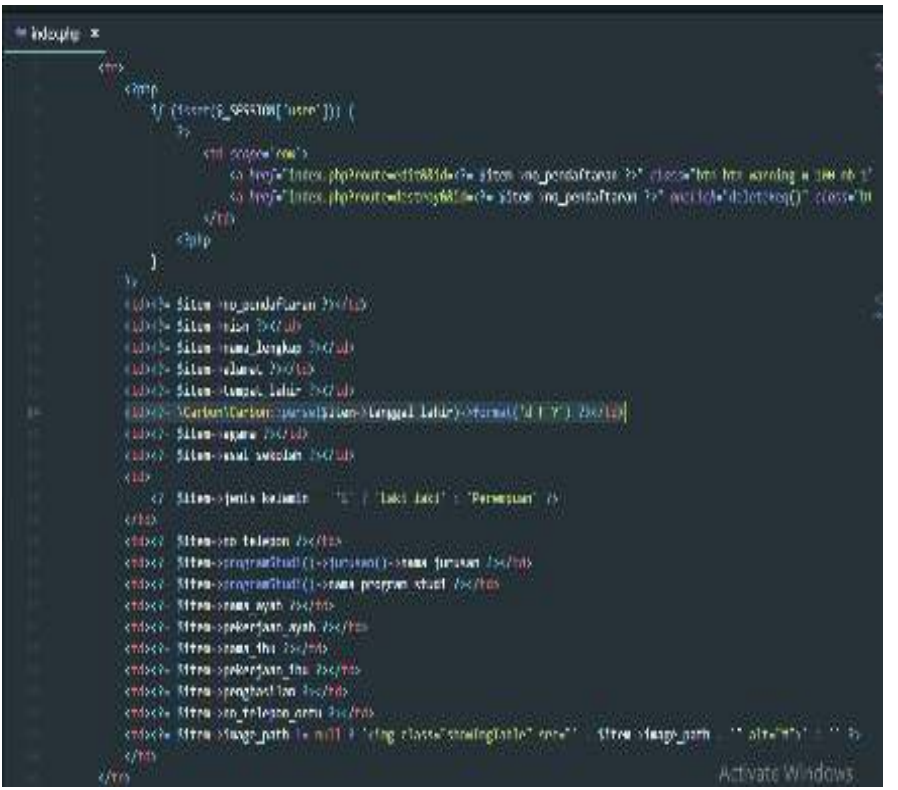
1. Menginstal *library carbon* lewat cmd.



Gambar 7.6 *Library* Carbon

2. Menambahkan *script code* di bawah ini pada *file* index.php untuk mengganti format tanggal.

```
<td><?=     \Carbon\Carbon::parse($item->tanggal_lahir)-
>format('d F Y')?></td>
```



Gambar 7.7 Penulisan *Script Library* Carbon

7.4 Soal Latihan

1. Installah paris ORM kedalam komputer/laptop.
2. Implementasikan lah paris ORM kedalam sebuah *web form input* yang terhubung kedalam *database* serta yang memuat tanggal.

3. Tambahkanlah *carbon* pada tanggal sehingga format tanggal berubah menjadi tanggal, bulan, tahun.
4. Buatlah ringkasan tentang ParisORM serta *carbon* ke dalam Google Docs, lengkap dengan contoh pembuatannya, dengan menggunakan bahasa sendiri!

DAFTAR PUSTAKA

Ariona, R. (2013). *Belajar HTML dan CSS: Tutorial Fundamental dalam mempelajari HTML & CSS*.

Duckett, J. (2011). *HTML & CSS design and build websites*. Canada: John Wiley & Sons.

Howe, S. (2014). *Learn to Code HTML & CSS: Develop & Style Websites*. Pearson Education.

Marlinda, [12]. (2004). *Sistem Basis Data*. Jogyakarta: Andi.

Muhardin, E. (2003). *PHP Programing Fundamental dan MYSQL*. ArtiVisi Intermedia.

Powell, T. A. (2010). *HTML & CSS: The Complete Reference Fifth Edition* (Fifth Edit). New York: Mc Graw Hill.

Pratama, Ha. (2016). *HTML Uncover: Panduan Belajar HTML untuk Pemula*. Duniailkom.

Robbins, J. N. (2012). *Learning Web Design: A Beginner's Guide To HTML, CSS, JavaScript And Web Graphics* (Fourth Edi). O'REILLY.

Wempen, F. (2011). *Step By Step HTML5*. Octal Publishing.

PEMROGRAMAN INTERNET

M. HENDRA SUNARYA **M. BAHIT**

HTML (*Hypertext Markup Language*) merupakan bahasa markah yang digunakan untuk membuat sebuah halaman web dengan tujuan untuk menampilkan atau berbagai informasi dalam sebuah web yang akses melalui internet. (Wempen, 2011). Semua halaman web yang sering Anda buka, seperti facebook.com, twitter.com, google.com akan ditampilkan menggunakan HTML. Jadi bisa dikatakan HTML adalah bahasa dasar untuk menampilkan halaman web pada *web browser* (Ariona, 2013).

HTML terdiri dari 3 komponen kata yaitu Hypertext, Markup dan Language. Kata Hypertext dari HTML berarti *"text"* yang tidak hanya berfungsi sebagai *text* biasa tetapi juga dapat berfungsi sebagai penghubung ke halaman lain atau yang sering dikenal dengan istilah *link*. Selain *text* dapat dijadikan *link*, sebuah gambar juga dapat dijadikan sebagai *link* untuk penghubung ke halaman lain (Pratama, 2016).

Penerbit Poliban Press
Redaksi :
Politeknik Negeri Banjarmasin, Jl. Brigjen H. Hasan Basry, Pangeran, Komp. Kampus ULM, Banjarmasin Utara
Telp : (0511)3305052
Email : press@poliban.ac.id