BAB 2
LANDASAN TEORI

2.1 Pernikahan

Menurut Undang-Undang Perkawinan, yang dikenal dengan Undang-Undang No.1 Tahun 1974, yang dimaksud dengan perkawinan adalah ikatan lahir batin antara seorang pria dan seorang wanita sebagai suami isteri dengan tujuan membentuk keluarga (rumah tangga) yang bahagia dan kekal berdasarkan Ketuhanan Yang Maha Esa. Perkawinan adalah pintu bagi bertemunya dua hati dalam naungan pergaulan hidup yang berlangsung dalam jangka waktu yang lama, yang di dalamnya terdapat berbagai hak dan kewajiban yang harus dilaksanakan oleh masing-masing pihak untuk mendapatkan kehidupan yang layak, bahagia, harmonis, serta mendapat keturunan. Perkawinan itu merupakan ikatan yang kuat yang didasari oleh perasaan cinta yang sangat mendalam dari masing-masing pihak untuk hidup bergaul guna memelihara kelangsungan manusia di bumi [11].

2.2 Multimedia

Multimedia adalah penggunaan komputer untuk menyajikan dan menggabungkan teks, suara, gambar, animasi, audio dan video dengan alat bantu (*tools*) dan koneksi (*link*) sehingga pengguna dapat melakukan navigasi, berinteraksi, berkarya dan berkomunikasi. Multimedia sering digunakan dalam dunia informatika. Selain dari dunia informatika, multimedia juga diadopsi oleh dunia *game*, dan juga untuk membuat *website*. Multimedia dimanfaatkan juga dalam dunia pendidikan dan bisnis. Di dunia pendidikan, multimedia digunakan sebagai media pengajaran, baik dalam kelas maupun secara sendiri-sendiri atau otodidak. Di dunia bisnis, multimedia digunakan sebagai media profil perusahaan, profil produk, bahkan sebagai media kios informasi dan pelatihan dalam sistem *e-learning* [10].

2.3 Jenis Multimedia

Jenis Multimedia Jenis multimedia terbagi menjadi tiga bagian [10], yaitu:

1. Multimedia interaktif

   Pengguna dapat mengontrol apa dan kapan elemen-elemen multimedia akan dikirimkan atau ditampilkan.
2. Multimedia hiperaktif

   Multimedia hiperaktif mempunyai suatu struktur dari elemen-elemen terkait dengan pengguna yang dapat mengarahkannya. Multimedia jenis ini mempunyai banyak tautan (*link*) yang menghubungkan elemen-elemen multimedia yang ada.
3. Multimedia linear

   Pengguna hanya menjadi penonton dan menikmati produk multimedia yang disajikan dari awal hingga akhir.

2.4 Aplikasi

Aplikasi adalah program yang memiliki aktivitas pemrosesan perintah yang diperlukan untuk melaksanakan permintaan pengguna dengan tujuan tertentu". Aplikasi dapat juga diartikan suatu program yang dibuat untuk tujuan tertentu sesuai dengan pengguna yang akan menggunakan suatu aplikasi tersebut. Jadi fungsi dari suatu aplikasi itu untuk membantu ataupun mempermudah kita sebagai manusia dalam melakukan tugas tertentu.[12].

2.5 Internet

Internet atau *Internetworking* secara umum didefinisikan sebagai jaringan komputer terbesar di dunia yang menghubungkan semua jaringan komputer yang ada (*Intranet*, *Wide Area Network*, *Metropolitan Area Network*, *Personal Area Network*, dan lain-lain) beserta dengan semua komputer, perangkat terhubung (*Smartphone, Switch*, *Router*, *Hub*, dan perangkat penghubung lainnya), serta pengguna komputer itu sendiri, ke dalam satu wadah jaringan komputer dunia [13].

2.6 WWW

World Wide Web (*WWW*) adalah suatu program yang ditemukan oleh Tim Berners-Lee pada tahun 1991. Awalnya Berners-Lee hanya ingin menemukan cara

untuk menyusun arsip-arsip risetnya. Untuk itu, beliau mengembangkan suatu sistem untuk keperluan pribadi. Sistem itu adalah program piranti lunak yang diberi nama *Enquire*. Dengan program itu, Berners-Lee berhasil menciptakan jaringan yang menautkan berbagai arsip sehingga memudahkan pencarian informasi yang dibutuhkan. Inilah yang kelak menjadi dasar dari sebuah perkembangan pesat yang dikenal sebagai *WWW*. *WWW* bekerja berdasarkan pada empat mekanisme berikut [13]:

1. Informasi disimpan di dalam dokumen yang sering kita sebut halaman *web*.
2. Halaman *web* adalah kumpulan *file* yang disimpan dalam komputer. Komputer tersebut dikenal dengan istilah *web server*.
3. Komputer yang mengakses isi dari halaman *web* disebut dengan *web client*.
4. *Web client* menampilkan halaman *web* dengan program yang dikenal dengan nama *web browser* seperti *Chrome, Firefox*, dan *Internet Explorer*.

2.7 API

API atau *Application Programming Interface* bukan hanya satu *set class* dan *method* atau fungsi dan *signature* yang sederhana. Akan tetapi *API* bertujuan untuk mengatasi "*clueless*" dalam membangun *software* yang berukuran besar, berawal dari sesuatu yang sederhana sampai ke yang kompleks dan merupakan perilaku komponen yang sulit dipahami. Secara sederhana dapat dipahami dengan membayangkan kekacauan yang akan timbul bila mengubah *database* atau skema *XML*. Perubahan ini dapat dipermudah dengan bantuan *API*.

Dari beberapa sumber yang didapat, dapat disimpulkan bahwa *API* adalah sekumpulan perintah, fungsi, *class* dan protokol yang memungkinkan suatu *software* berhubungan dengan *software* lainnya. Tujuan dari *API* adalah untuk menghilangkan "*clueless*" dari sistem dengan cara membuat blok besar yang terdiri dari *software* di seluruh dunia dan menggunakan kembali perintah, fungsi, *class*, atau protokol yang mereka atau *API* miliki. Dengan cara ini, *programmer* tidak perlu lagi membuang waktu untuk membuat dan menulis infrastruktur sehingga akan menghemat waktu kerja dan lebih efisien [14].

2.8 Pemrograman Berorientasi Objek (*OOP*)

Object Oriented Programming (*OOP*) atau pemograman berorientasi Objek adalah suatu cara baru dalam berfikir serta berlogika dalam menghadapi masalah-masalah yang akan dicoba-atasi dengan bantuan komputer. *OOP* tidak seperti pendahulunya (pemograman terstruktur), mencoba melihat permasalahan lewat pengamatan dunia nyata di mana setiap objek adalah entitas tunggal yang memiliki kombinasi struktur data dan fungsi tertentu. Ini kontras dengan pemograman terstruktur dimana struktur data dan fungsi didefinisikan secara terpisah dan tidak berhubungan secara erat [15].

Secara spesifik, pengertian "berorientasi objek" berarti bahwa kita mengorganisasi perangkat lunak sebagi kumpulan dari objek tertentu yang memiliki struktur data dan perilakunya. Hal ini yang membedakan dengan pemrograman konvensional di mana struktur data dan perilakunya hanya berhubungan secara terpisah [16].

Pada perkembangannya, filosofi *OOP* menciptakan sinergi yang luar biasa sepanjang siklus pengembangan perangkat lunak (perancanaan, analisis, perancangan, implementasi, serta pengujian) sehingga dapat diterapkan pada perancangan sistem secara umum; menyangkut perangkat lunak, perangkat keras serta sistem informasi secara keseluruhan [15].

Objek adalah konsep atau abstraksi tentang sesuatu yang memiliki arti bagi aplikasi yang akan kita kembangkan. Objek biasanya adalah kata benda, namun objek dalam konteks *OOP* bukan hanya objek nyata yang bisa diraba dan dilihat secara kasat mata seperti mobil, pesawat terbang, sapi, kuda, gitar, buku, *tape-recorder*, komputer dan sebagainya, namun juga menyangkut entitas-entitas konseptual seperti rumusan persamaan kuadrat, *liberalism*, marxisme, dan sebagainya [15].

Setiap objek adalah nyata dan dapat dibedakan satu dari yang lainnya. Sekalipun 2 objek bernama sama, namun ia tetap dapat dipisahkan. Walaupun satu objek adalah kembar misalnya, kita tetap bisa membedakan satu individu terhadap individu yang lainnya. Misalnya, untuk objek kembar, sesuatu yang

membedakannya adalah (misalnya) namanya, sehingga objek yang satu memiliki identitas yang berbeda dengan yang lainnya.

Maka disimpulkan metodologi berorientasi objek merupakan suatu pembangunan perangkat lunak yang mengorganisasikan perangkat lunak sebagai kumpulan objek yang berisi data dan operasi yang diberlakukan terhadapnya dan metodologi berorientasi objek merupakan suatu cara bagaimana perangkat lunak dibangun melalui pendekatan objek secara sistematis. Metode berorientasi objek didasarkan pada penerapan prinsip-prinsip pengelolaan kompleksitas. Metode berorientasi objek meliputi rangkaian aktivitas analisis berorientasi objek, perancangan berorientasi objek, pemrograman berorientasi objek dan pengujian berorientasi objek.

2.9 *UML* (*Unified Modeling Languange)*

Unified Modeling Languange (UML) adalah keluarga notasi grafis yang didukung oleh meta-model tunggal, yang membantu pendeskripsian dan desain sistem perangkat lunak, khususnya sistem yang dibangun menggunakan pemrograman berorientasi objek (*OOP*). Definisi ini merupakan definisi yang sederhana. Pada kenyataannya, pendapat orang-orang tentang *UML* berbeda satu sama lain. Hal ini dikarenakan oleh sejarahnya sendiri dan oleh perbedaan persepsi tentang apa yang membuat sebuah proses rancang-bangun perangkat lunak efektif [17].

Unified Modeling Language (*UML*) adalah salah satu alat bantu yang sangat handal di dunia pengembangan sistem yang berorientasi objek. Hal ini disebabkan karena *UML* menyediakan bahasa pemodelan *visual* yang memungkinkan bagi pengembang sistem untuk membuat cetak biru atas visi mereka dalam bentuk yang baku, mudah dimengerti serta dilengkapi dengan mekanisme yang efektif untuk berbagi (*sharing*) dan mengkomunikasikan rancangan mereka dengan yang lain [18].

Secara umum *UML* merupakan ‘bahasa’ untuk visualisasi, sepesifikasi, konstruksi, serta dokumentasi. Dalam kerangka visualisasi, para pengembang menggunakan *UML* sebagai suatu cara untuk mengkomunikasikan idenya kepada

para pemrogram serta calon pengguna sistem/perangkat lunak. Dengan adanya 'bahasa' yang bersifat standar, komunikasi perancang dengan pemrogram serta calon pengguna diharapkan menjadi mulus [17].

Dalam kerangka spesifikasi, *UML* menyediakan model-model yang tepat, tidak mendua-arti (ambigu), serta lengkap. Secara khusus, *UML* menspesifikasi langkah-langkah penting dalam pengambilan keputusan analisis, perancangan, serta implementasi dalam sistem yang sangat bernuansa perangkat lunak (*software intensive system*). Dalam hal itu, *UML* bukanlah merupakan Bahasa pemrograman tetapi model-model yang tercipta berhubungan langsung dengan berbagai macam Bahasa pemrograman perorientasi objek, katakanlah *Java, Borland Delphi, Visual BASIC, C++*, dan lain-lain [15].

UML lahir dari penggabungan banyak Bahasa pemodelan grafis berorientasi objek yang berkembang pesat pada akhir 1980-an dan awal 1990-an. Sejak kehadirannya pada tahun 1997, *UML* menghancurkan Menara Babel tersebut menjadi sejarah. Pada intinya peran *UML* dalam pengembangan perangkat lunak, orang-orang memiliki cara-cara yang berbeda dalam penggunaanya, perbedaan-perbedaan yang masih dibawa dari bahasa-bahasa pemodelan grafis lain. Perbedaan-perbedaan ini mengakibatkan perselisihan yang panjang dan keras tentang bagaimana *UML* seharusnya digunakan.

Berdasarkan pemaparan mengenai *Unified Modeling Languange* (UML) dari beberapa sumber referensi, maka dapat disimpulkan *UML* merupakan alat bantu dalam melakukan pemodelan yang saling berhubungan secara langsung dalam pembangunan sebuah sistem agar lebih efektif.

2.9.1 Use Case Diagram

Use Case adalah deskripsi fungsi dari sebuah sistem dari perspektif pengguna. *Use Case* bekerja dengan cara mendeskripsikan tipikal interaksi antara pengguna sebuah sistem dengan sistemnya sendiri melalui sebuah cerita bagaimana sebuah sistem dipakai [18].

Use Case diagram digunakan untuk menggambarkan analisis kebutuhan dari sistem dari *level* atas melalui fungsionalitas dari sistem dan interaksi diantara para aktor. Aktor adalah sesuatu yang berinteraksi dengan sistem.

Secara umum, tujuan dari *use case diagram* adalah sebagai berikut [18]:

1. Digunakan untuk mengumpulkan kebutuhan dari sebuah sistem.
2. Untuk mendapatkan pandangan dari luar sistem.
3. Untuk mengidentifikasi faktor yang mempengaruhi sistem baik internal maupun eksternal.

2.9.2 Activity Diagram

Activity Diagram adalah bagian penting dari *UML* yang menggambarkan aspek dinamis dari sistem. Logika *procedural*, proses bisnis, dan aliran kerja suatu bisnis bisa dengan mudah dideskripsikan dalam *activity diagram*. *Activity diagram* memiliki peran seperti halnya *flowchart*, akan tetapi perbedaannya adalah *activity diagram* bisa mendukung perilaku paralel [18].

Tujuan dari *activity diagram* adalah untuk menangkap tingkah laku dinamis dari sistem dengan cara menunjukkan aliran pesan dari satu aktivitas ke aktivitas lainnya. Secara umum tujuan *activity diagram* adalah sebagai berikut [18]:

1. Menggambarkan aliran aktivitas dari sistem.
2. Menggambarkan urutan aktivitas dari satu aktivitas ke aktivitas lainnya.
3. Menggambarkan paralelisme, percabangan dan aliran konkuren dari sistem.

2.9.3 Class Diagram

Class Diagram adalah diagram statis yang mewakili pandangan statis dari suatu aplikasi. *Class diagram* tidak hanya digunakan untuk memvisualisasikan, menggambarkan, dan mendokumentasikan berbagai aspek sistem tetapi juga untuk membangun kode eksekusi (*executable code*) dari aplikasi perangkat lunak. C*lass diagram* menunjukan koleksi kelas, antarmuka, asosiasi, kolaborasi, dan *constraint*. *Class diagram* juga dikenal sebagai *diagram structural* [18].

Tujuan dari *class diagram* adalah untuk memodelkan pandangan statis suatu aplikasi. Secara lebih rinci tujuannya adalah sebagai berikut [18]:

1. Analisis dan desain pandangan statis aplikasi.
2. Menjelaskan tanggung jawab suatu sistem.
3. Basis untuk *diagram* komponen dan penyebaran (*deployment*).
4. *Forward and reverse engineering.*

2.9.4 Sequence Diagram

Sequence Diagram digunakan untuk menggambarkan perilaku pada sebuah skenario. Diagram ini menunjukan sejumlah contoh obyek dan *message* (pesan) yang diletakkan diantara obyek-obyek ini di dalam *use case*. Komponen utama *sequence diagram* terdiri atas obyek yang dituliskan dengan kotak segi empat bernama. *Message* diwakili oleh garis dengan tanda panah dan waktu yang ditunjukkan dengan *progress vertical* [18].

2.10 Metode Pengujian Sistem

Pengujian perangkat lunak merupakan tahapan untuk menemukan kesalahan-kesalahan dan kekurangan-kekurangan pada perangkat lunak yang dibangun sehingga bisa diketahui apakah perangkat lunak tersebut telah memenuhi kriteria sesuai dengan tujuan atau tidak. Adapun metode pengujian yang digunakan pada perangkat lunak ini adalah metode *black box*. Pengujian *black box* berfokus pada persyaratan fungsional perangkat lunak. Metode pengujian *black box* ini tediri dari dua tahapan pengujian yaitu tahapan pengujian *alpha* dan tahapan pengujian *beta* [19].

2.10.1 Pengujian *Alpha*

Pengujian *alpha* merupakan pengujian fungsional yang diadakan di lingkungan pembangunan oleh sekumpulan pengguna yang akan menggunakan perangkat lunaknya. Pihak pembangunan mendampingi serta mencatat kesalahan kesalahan maupun permasalahan yang dirasakan oleh pengguna [19].

2.10.2 Pengujian *Black Box*

Pengujian yang dilakukan hanya mengamati hasil eksekusi melalui data uji dan memeriksa fungsional dari perangkat lunak. Pengujian ini dianalogikan seperti melihat suatu kotak hitam yang hanya bisa dilihat penampilan luarnya saja, tanpa tahu ada apa dibalik bungkus hitamnya [19]. *Black box* testing melakukan evaluasi untuk menemukan kesalahan dalam kategori berikut:

1. Fungsi tidak benar atau hilang.
2. Kesalahan interface atau antarmuka.
3. Kesalahan dalam struktur data atau akses database eksternal.
4. Kesalahan kinerja atau perilaku dan kesalahan inisialisasi dan terminasi.

2.10.3 Pengujian *Beta*

Pengujian *beta* merupakan pengujian langsung kepada pengguna untuk mencoba aplikasi yang baru. Pengujian *beta* diadakan di lingkungan *live* (sebenarnya), dalam Pengujian *beta end user* mencatat kemudian menyampaikan pada pihak *developer*. Pengujian dilakukan pada satu atau lebih pelanggan oleh pemakai akhir perangkat lunak dalam lingkungan yang sebenarnya, pengembang biasanya tidak ada pada pengujian ini. Pelanggan merekam semua masalah (*real* atau *imajiner*) yang ditemui selama pengujian dan melaporkan pada pengembang pada *interval* waktu tertentu. Pengujian bertujuan untuk meningkatkan jumlah para pemakai di masa yang akan datang [19].

2.10.4 Skala Likert

Skala data yang digunakan untuk mengukur variabel yang didapat, menggunakan skala likert. Skala likert adalah skala respon psikometri terutama digunakan dalam kuesioner untuk mendapatkan preferensi responden atas sebuah pernyataan atau serangkaian laporan [20]. Setelah menyelesaikan definisi operasional variabel maka langkah selanjutnya menyusun item-item. Sebuah skala menjadi penting untuk mengukur derajat pendapat dan data kuantitatif berarti analisis relatif mudah dilakukan. Prinsip pengukuran sikap yaitu meminta orang untuk menanggapi serangkaian pernyataan tentang suatu topik. Sejauh mana

mereka setuju dengan memasuki komponen kognitif dan afektif. Data yang telah terkumpul melalui angket, kemudian diolah kedalam bentuk kuantitatif, yaitu dengan sebuah kontinum dari sangat setuju sampai sangat tidak setuju yang dapat dilihat pada Tabel 2.1.

Tabel 2.1 Penilaian Skala Likert

Jawaban	Skor
Sangat Setuju	5
Setuju	4
Netral	3
Kurang Setuju	2
Tidak Setuju	1

Kemudian dengan teknik pengumpulan data kuisioner, maka instrument tersebut misalnya diberikan kepada 30 orang yang diambil secara random. Dari 30 orang tersebut setelah dilakukan analisis didapat hasil sebagai berikut:

1. Sebanyak 15 orang menjawab Sangat Setuju (SS)
2. Sebanyak 10 orang menjawab Setuju (S)
3. Sebanyak 2 orang menjawab Netral (N)
4. Sebanyak 2 orang menjawab Kurang Setuju (KS)
5. Sebanyak 1 orang menjawab Tidak Setuju (TS)

Berdasarkan data tersebut 25 orang (15 + 10) atau 83.3% stakeholder menjawab setuju dan sangat setuju.

Data Interval tersebut kemudian dianalisis dengan menghitung rata-rata jawaban berdasarkan skoring setiap jawaban dari responden. Berdasarkan skor yang telah didapat dari analisis sebelumnya dapat dihitung sebagai berikut:

Jumlah skor untuk 15 orang yang menjawab SS = 15x5 = 75

Jumlah skor untuk 10 orang yang menjawab S = 10x4 = 40

Jumlah skor untuk 2 orang yang menjawab N = 2x3 = 6

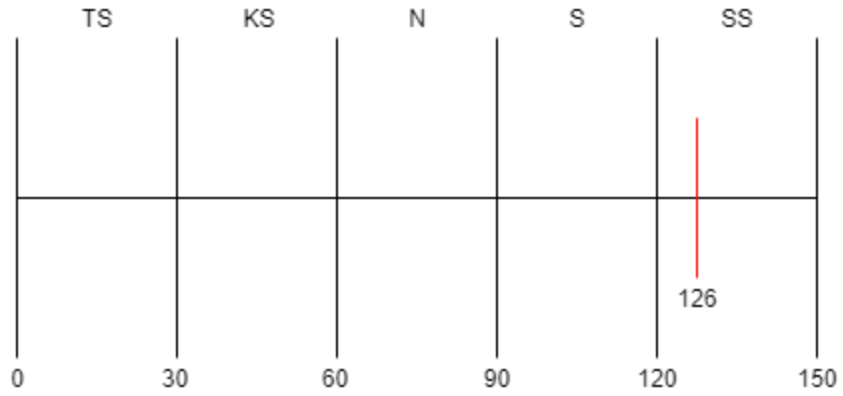
Jumlah skor untuk 2 orang yang menjawab KS = 2x2 = 4

Jumlah skor untuk 1 orang yang menjawab TS = 1x1 = 1

Jumlah Total Nilai = 126

Jumlah ideal (kriterium untuk seluruh item = 5x30 = 150 (seandainya semua menjawab SS). Jumlah skor yang diperoleh dari penelitian = 126. Jadi berdasarkan data itu maka tingkat persetujuan stakeholder terhadap penggunaan media pembelajaran interaktif = (126:150) x 100% = 84% dari yang diharapkan (100%).

Secara kontinum dapat dilihat seperti pada Gambar 2.16.



Gambar 2.1 Pengolahan Hasil Kuesioner Secara Kontinum

Jadi berdasarkan data yang diperoleh dari 30 responden maka rata-rata 126 terletak pada daerah mendekati setuju.

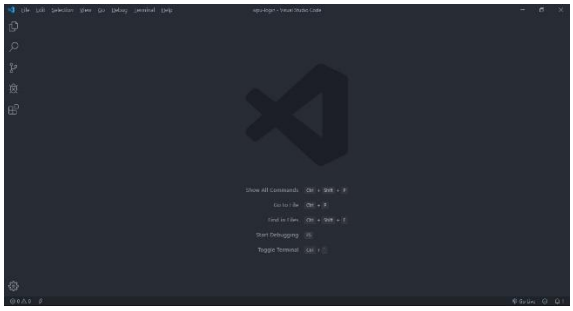
Alasan mengapa menggunakan skala likert untuk mengukur variabel adalah sebagai berikut:

1. Skala likert memudahkan responden untuk menjawab kuesioner apakah setuju atau tidak setuju.
2. Skala likert mudah digunakan dan mudah dipahami oleh responden
3. Skala likert secara visual lebih menarik dan mudah diisi oleh responden.

2.11 *Tools* Yang Digunakan

Tools yang digunakan untuk membangun aplikasi multimedia interaktif meliputi *Visual Studio Code, HTML, Javascript, CSS, Bootstrap, PHP, Framework CodeIgniter, MySQL, AddToAny Plugin, Google Maps API, Twitter API*.

2.11.1 Visual Studio Code



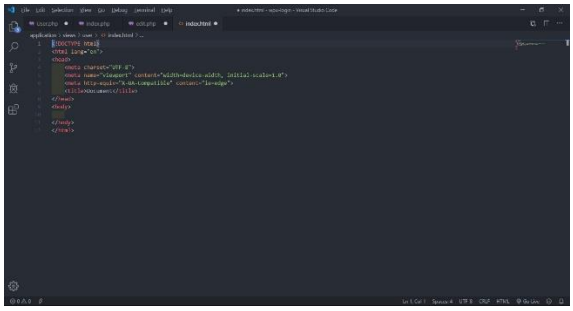
Gambar 2.2 Tampilan Awal *Visual Studio Code*

Visual Studio adalah *integrated development environment* (*IDE*) yang dikembangkan oleh *Microsoft* untuk mempermudah *software developer* mengembangkan aplikasi pada *platform* milik *Microsoft*. *Visual studio* 2015 adalah versi stabil terbaru saat buku ini ditulis dan sedang dikembangkan *visual studio* 2017. *Visual studio* dapat digunakan untuk mengembangkan aplikasi *mobile, web, desktop* dan *cloud*. Bahasa yang didukung oleh *visual studio* 2015 adalah *visual basic, C#, C++, python, javascript* dan masih banyak lagi. Tetapi *visual studio* 2015 hanya dapat digunakan pada sistem operasi *Microsoft Windows*. Tetapi saat ini *Microsoft* telah mengembangkan *visual studio code*.

Visual studio code adalah *source code editor multiplatform* yang dapat digunakan pada sistem operasi *Windows, Linux* dan *Mac OSX*. *Visual studio code* juga mendukung banyak bahasa pemrograman seperti halnya *visual studio* 2015 ditambah bahasa pemrograman *PHP, Node.js* dan lain-lain. Fitur-fitur utama dari *Visual Studio Code* adalah [21]:

1. *Intelligent code completion*, fitur ini akan membantu *software developer* untuk melengkapi *variable, method* dan modul yang ditulis.
2. *Streamlined debugging*, fitur ini berfungsi untuk melakukan *debug* terhadap kode yang ditulis.
3. *Linters, multi-cursor editing*, parameter *hints*.
4. *Code navigation*.
5. *Refactoring* dukungan akses *Git*.

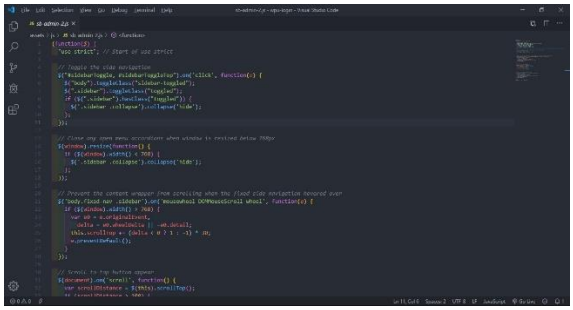
2.11.2 HTML



Gambar 2.3 Contoh Kode *HTML*

HTML (*Hypertext Markup Language*) adalah sebuah bahasa pemrograman yang berbentuk skrip-skrip yang berguna untuk membuat sebuah halaman *web*. *HTML* dapat dibaca oleh berbagai *platform* seperti: *Windows*, *Linux*, *Macintosh*. Kata "*Markup Language*" pada *HTML* menunjukkan fasilitas yang berupa tanda tertentu dalam skrip *HTML* dimana kita bisa mengatur judul, garis, tabel, gambar, dan lain-lain dengan perintah yang telah ditentukan pada elemen *HTML*. *HTML* sendiri dikeluarkan oleh *W3C (Word Wide Web Consortin*), setiap terjadi perkembangan *level HTML* harus dievakuasi ketat dan disetujui oleh *W3C* [22].

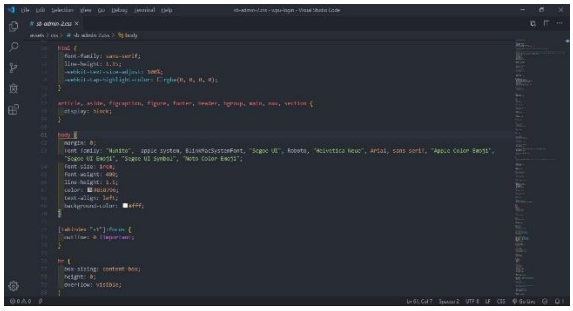
2.11.3 JavaScript



Gambar 2.4 Contoh Kode *JavaScript*

JavaScript merupakan modifikasi dari bahasa *c++* dengan pola penulisan yang lebih sederhana. *Interpreter* bahasa ini sudah disediakan ASP ataupun *internet explorer*. Kelebihan *Java Script* adalah berinteraksi dengan *HTML*, ini membolehkan pembuat *web* untuk memasukkan *web* mereka dengan kandungan-kandungan yang dinamik, menukar warna background, menukar *banner*, efek *mouse, menu* interaktif dan sebagainya [22].

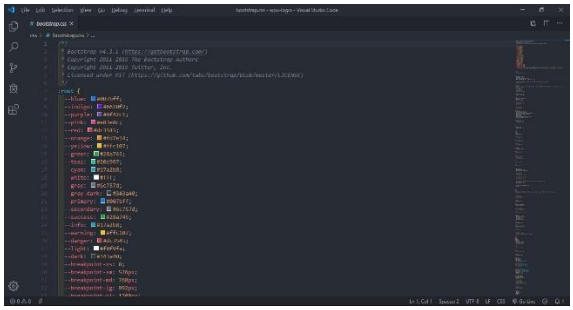
2.11.4 CSS



Gambar 2.5 Contoh Kode *CSS*

CSS adalah singkatan dari *Cascading Style-Sheet*, yaitu sebuah pengembangan atas kode *HTML* yang sudah ada sebelumnya. Dengan *CSS*, bisa menentukan sebuah struktur dasar halaman *web* secara lebih mudah dan cepat, serta irit *size* [22].

2.11.5 Bootstrap



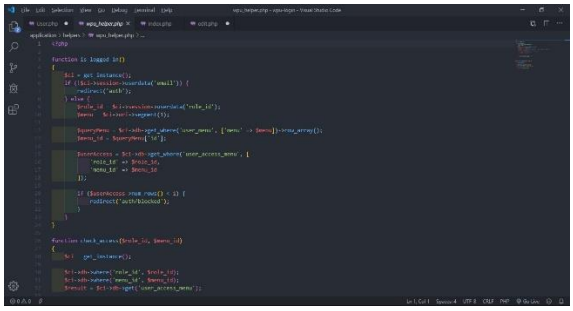
Gambar 2.6 Contoh Kode *Bootstrap*

Bootstrap merupakan *Framework* ataupun *Tools* untuk membuat aplikasi *web* ataupun situs *web responsive* secara cepat, mudah dan gratis.

Bootstrap terdiri dari *CSS* dan *HTML* untuk menghasilkan *Grid*, *Layout*, *Typography*, *Table*, *Form*, *Navigation*, dan lain-lain. Di dalam *Bootstrap* juga sudah terdapat *jQuery plugins* untuk menghasilkan komponen *UI* yang cantik seperti *Transitions*, *Modal*, *Dropdown*, *Scrollspy*, *Tooltip*, *Tab*, *Popover*, *Alert*, *Button*, *Carousel* dan lain-lain.

Dengan bantuan *Bootstrap*, kita bisa membuat *responsive website* dengan cepat dan mudah dan dapat berjalan sempurna pada *browser-browser* populer seperti *Chrome*, *Firefox*, *Safari*, *Opera* dan *Internet Explorer* [23].

2.11.6 PHP



Gambar 2.7 Contoh Kode *PHP*

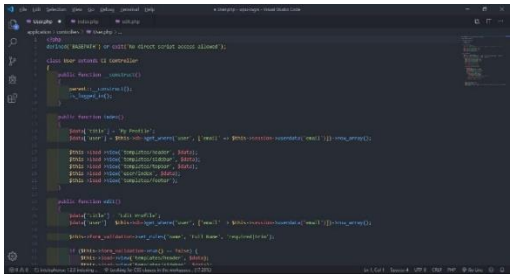
PHP adalah bahasa pemrograman *web* atau *scripting language* yang dijalankan di *server*. *PHP* dibuat pertama kali oleh Rasmus Lerdorf, yang pada awalnya dibuat untuk menghitung jumlah pengunjung pada *homepage*-nya. Pada waktu itu *PHP* bernama *FI* (*Form Interpreter*). Pada saat tersebut *PHP* adalah sekumpulan *script* yang digunakan untuk mengolah data *form* dari *web*. Perkembangan selanjutnya adalah Rasmus melepaskan kode sumber tersebut dan menamakannya *PHP/FI*, pada saat tersebut kepanjangan dari *PHP/FI* adalah *Personal Home Page/Form Interpreter*. Pelepasan kode sumber ini menjadi *open source*, maka banyak *programmer* yang tertarik untuk ikut mengembangkan *PHP*.

Pada tahun 1997 sebuah perusahaan bernama Zend, menulis ulang *interpreter PHP* mejadi lebih bersih, lebih baik dan lebih cepat. Kemudian pada Juni 1998 perusahaan tersebut merilis *interpreter* baru untuk *PHP* dan meresmikan nama rilis tersebut menjadi *PHP* 3.0. Pada pertengahan tahun 1999, Zend merilis *interpreter PHP* baru dan rilis tersebut dikenal dengan *PHP* 4.0. *PHP* 4.0 adalah versi *PHP* yang paling banyak dipakai. Versi ini banyak dipakai sebab versi ini mampu dipakai untuk membangun aplikasi *web* kompleks tetapi tetap memiliki kecepatan proses dan stabilitas yang tinggi.

Pada Juni 2004 Zend merilis *PHP 5.0*. Versi ini adalah versi mutakhir dari *PHP*. Dalam versi ini, inti dari *interpreter PHP* mengalami perubahan besar. Dalam versi ini juga dikenalkan model pemrograman berorientasi objek baru untuk menjawab perkembangan bahasa pemrograman kearah pemrograman berorientasi objek. Hal yang menarik yang didukung oleh *PHP* adalah kenyataan bahwa *PHP*

bisa digunakan untuk mengakses berbagai macam *database* seperti *Access*, *Oracle*, *MySQL*, dan lain-lain [22].

2.11.7 Framework CodeIgniter



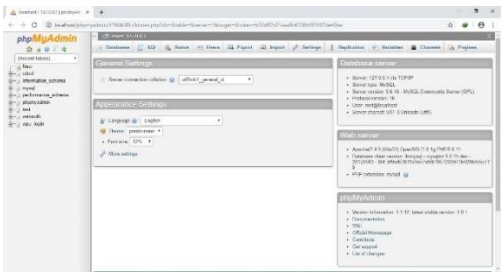
Gambar 2.8 Contoh Kode *CodeIgniter*

Framework adalah abstraksi di dalam sebuah perangkat lunak yang menyediakan fungsi yang *generic* sehingga dapat dirubah oleh kode yang dibuat *user* sehingga dapat menyediakan perangkat lunak untuk aplikasi tertentu.

Metode *MVC* adalah sebuah arsitektur yang dapat diimplementasikan secara bebas dengan atau tanpa bahasa pemrograman berorientasi objek. Dengan demikian metode *MVC* dapat diimplementasikan dalam sebuah *framework*.

CodeIgniter merupakan sebuah *framework* pemrograman *web* dengan menggunakan bahasa *php*. *Framework* ini ditulis dengan menggunakan bahasa *php* versi 4 dan versi 5 oleh Rick Ellislab yang menjadi *CEO Ellislab, Inc*. dan dipublikasikan dengan lisensi di bawah *Apache/BSD Open Source*. Jadi *CodeIgniter* adalah *framework php* dan *Open Source* [24].

2.11.8 MySQL



Gambar 2.9 Tampilan phpmyadmin *MySQL*

MySQL (*My Structured Query Language*) atau yang biasa dibaca mai-se-kuel adalah sebuah program pembuat dan pengelola *database* atau yang sering disebut

dengan *DBMS* (*DataBase Management System*), sifat dari *DBMS* ini adalah *Open Source*. *MySQL* sebenarnya produk yang berjalan pada *platform Linux*, dengan adanya perkembangan dan banyaknya pengguna, serta lisensi dari *database* ini adalah *Open Source*, maka para pengembang kemudian merilis versi *Windows*.

Selain itu *MySQL* juga merupakan program pengakses *database* yang bersifat jaringan, sehingga dapat digunakan untuk aplikasi Multi *User* (Banyak Pengguna). Kelebihan lain dari *MySQL* adalah menggunakan bahasa *query* (permintaan) standar *SQL* (*Structured Query Language*). Sebagai sebuah program penghasil *database*, *MySQL* tidak mungkin berjalan sendiri tanpa adanya sebuah aplikasi pengguna (*interface*) yang berguna sebagai program aplikasi pengakses *database* yang dihasilkan. *MySQL* dapat didukung oleh hampir semua program aplikasi baik yang *Open Source* seperti *PHP* maupun yang tidak *Open Source* yang ada pada *platform* windows seperti *Visual Basic*, *Delphi* dan lainnya [22].

2.11.9 AddToAny Share Button

Tombol *plugin* berbagi *AddToAny* meningkatkan lalu lintas & keterlibatan dengan membantu orang membagikan posting dan halaman ke layanan apa pun. Layanan mencakup *Facebook*, *Twitter*, *Pinterest*, *Google*, *WhatsApp*, *LinkedIn*, *Tumblr*, *Reddit*, dan lebih dari 100 situs dan aplikasi media sosial lainnya [25].

2.11.10 Google Maps API

Google Maps adalah layanan aplikasi dan teknologi peta berbasis *web* yang disediakan oleh *Google* secara gratis (bukan untuk kepentingan komersial), temasuk di dalamnya *website Google Maps* (http://maps.google.com), *Google Ride Finder*, *Google Transit*, dan peta yang dapat disisipkan pada *website* lain melalui *Google Maps API* [26]. Beberapa *Syntax* yang sering digunakan dalam *Google Map* adalah berikut [27]:

1. *Google.maps.LatLng*

   Merupakan sintaks yang digunakan untuk menunjuk pada lokasi di peta. *LatLng* memiliki banyak kegunaan pada *GoogleMaps API*. Contohnya:

google.maps.Marker menggunakan *LatLng* dalam *constructor* dan meletakkan *marker* penanda pada lokasi geografis di peta.

```
Var myLatLng = new google.maps.LatLng(my Latitude,my Longitude)
```

2. Google.maps.Map

Merupakan *class.JavaScript* yang merepresentasikan sebuah peta. *Object* pada *class* didefinisikan dalam sebuah peta di halaman *website*. Secara sederhana dapat dikatakan bahwa *Map()* membuat sebuah *object* peta dan memasukannya ke dalam *mapDiv*.

```
Var map = new google.maps.Maps(document.getElementById
"map_canvas"), myOptions);
```

3. Google.maps.Marker

Membuat sebuat *marker* atau penanda pada pilihan tertentu. Bila sebuah peta spesifik, penanda diletakan pada peta pada saat *construction*. Perhatikan bahwa penanda harus diatur agar *pinpoint*.

```
Var marker = new google.maps.Marker(opts?:MarkerOptions);
```

4. Google.maps.event.addListener

Setiap *object Google Maps API* membuat sejumlah nama *event*. Program yang ingin menggunakan *event* tertentu akan memanggil *event listener* untuk

event tersebut dan menjalankan kode ketika *event* diterima dengan mendaftarkannya memalui *addListener()*. Secara sederhana, *addListener()* memanggil nama-nama suatu *object* (misalnya: *marker* atau *map* atau *sizer*) dan menjalankan ketika suatu kejadian terjadi (misalnya ketika diklik).

```
var handle = google.maps.event.addListener(Object, UserEvent,
function(event) { //Do something });
```

5. *GetZoom() dan SetZoom()*

GetZoom() mengembalikan sebuah nilai yang mengidentifikasikan nilai dari *zoom level* yang sekarang. *setZoom(zomLevel:number)* mengatur tingkat *zoom* peta.

```
Var zoomLevel = map.getZoom(); map.setZoom(12);
```

2.11.11 Twitter API

Pada tingkat tinggi, *API* merupakan cara program komputer berkomunikasi satu sama lain agar mereka dapat meminta dan menyajikan informasi. Ini dilakukan dengan mengizinkan aplikasi perangkat lunak memanggil apa yang disebut sebagai *endpoint*: alamat yang terkait dengan informasi jenis tertentu yang disediakan. *Twitter* mengizinkan akses ke bagian dari layanan melalui *API* untuk memungkinkan orang-orang membangun perangkat lunak yang terintegrasi dengan *Twitter* seperti solusi yang membantu sebuah perusahaan menjawab umpan balik pelanggan di *Twitter* [28].

Data *Twitter* berbeda dari data yang dibagi oleh kebanyakan *platform* sosial lain karena data tersebut mencerminkan informasi yang dipilih pengguna untuk dibagikan ke publik. *Platform API Twitter* menyediakan akses luas ke data *Twitter* publik yang telah dipilih pengguna untuk dibagikan ke dunia. *Twitter* juga mendukung *API* yang memungkinkan pengguna mengelola informasi *Twitter* mereka yang non-publik (mis. *Direct Message*) dan memberikan informasi ini ke pengembang yang telah diizinkan pengguna untuk melakukannya [28].