BAB II
TINJAUAN PUSTAKA

2.1 Studi Literatur

Pada penelitian terkait ini, penulis mengkaji penelitian terdahulu mengenai sistem aplikasi yang dilakukan oleh penelitian-penelitian sebelumnya yang dapat menjadi dasar dari penelitian yang dilakukan. Beberapa penelitian mengenai penerapan *sms gateway* pada sekolah di antaranya adalah sebagai berikut:

1. Penelitian yang dilakukan oleh (Wihasto & Iqbal, 2016) dengan judul "Sistem Aplikasi *SMS Gateway* Sebagai Sarana Informasi Pada SMK Dr Tjipto Semarang". Penelitian tersebut dilakukan dengan tujuan untuk merancang sebuah sistem yang dapat memberikan informasi kehadiran siswa secara langsung kepada orang tua atau wali siswa serta berbasis autoreply agar siswa maupun wali murid SMK Dr Tjipto Semarang dapat mendapatkan informasi secara langsung. Keterkaitan antara penelitian tersebut dengan topik penelitian yang akan dilakukan dalam penelitian ini yaitu dalam hal implementasi penerapan *SMS Gateway* untuk melakukan proses *broadcast* suatu informasi kepada orang tua atau wali siswa.
2. Penelitian yang dilakukan oleh (Yusuf & Susanto, 2010) dengan judul "Pemanfaatan *SMS Gateway* Untuk Absensi Sekolah Siswa". Penelitian tersebut bertujuan untuk membuat sistem absensi siswa yang dapat dengan mudah diakses oleh orang tua siswa, sehingga orang tua siswa tidak harus datang ke sekolah untuk mengawasi, tetapi mereka cukup mengirimkan *SMS* dan sistem akan menjawabnya, apakah siswa yang bersangkutan masuk, alfa, atau bolos. Pemantauan dapat dilakukan oleh orang tua di mana pun dan kapan pun juga.
3. Penelitian yang dilakukan oleh (Akbar dkk., 2013) dengan judul "Perancangan Aplikasi *SMS Gateway* Pelaporan Nilai Siswa". Penelitian ini bertujuan untuk membuat sistem aplikasi yang bertujuan untuk mempermudah penyebaran informasi data nilai sekolah dengan menggunakan SMS, sehingga memungkinkan wali murid siswa untuk untuk dapat mengakses informasi dari

sekolah secara efektif dan memberikan kemudahan tanpa harus datang langsung ke sekolah. Cara kerja aplikasi ini yaitu wali murid siswa mengirimkan SMS dengan format tertentu ke nomor server tertentu, setelah SMS yang dikirimkan wali murid diterima oleh aplikasi selanjutnya aplikasi akan mengirimkan balasan SMS secara otomatis terkait dengan informasi yang diinginkan oleh wali murid siswa tersebut.

4. Penelitian yang dilakukan oleh (Fachrizal, 2015) dengan judul "*Prototype* Sistem Informasi Pengelolaan Akademik Berbasis SMS *Gateway* di SMA Negeri 22 Bandung". Penelitian ini bertujuan untuk mempermudah bagi siswa untuk mendapatkan informasi akademik dengan cepat, kapanpun dan di manapun. Untuk mengakses informasi nilai UTS, UAS dan nilai akhir dapat diperoleh dengan mengirimkan SMS ke sekolah sehingga siswa dapat mengetahui informasi di mana pun.

2.2 Aplikasi

Aplikasi adalah suatu unit perangkat lunak yang dibuat untuk melayani kebutuhan akan beberapa aktivitas seperti sistem perniagaan, *game* pelayanan masyarakat, periklanan, atau semua proses yang hampir dilakukan manusia (Pramana, 2012). Sedangkan menurut (Yuhefizar, 2012), aplikasi merupakan suatu program yang dikembangkan untuk memenuhi kebutuhan pengguna dalam menjalankan pekerjaan tertentu.

Berdasarkan kedua pendapat ahli di atas, maka dapat disimpulkan bahwa aplikasi merupakan sebuah program perangkat lunak yang dibuat dengan tujuan memenuhi kebutuhan dari aktivitas yang dilakukan oleh pengguna serta mempermudah pekerjaan dari pengguna tersebut.

2.3 SMS (*Short Message Service*)

Menurut (Wahana Komputer, 2010) mengartikan, *SMS (Short Message Service)* adalah merupakan salah satu layanan pesan teks yang dikembangkan dan distandarisasi oleh suatu badan bernama ETSI (*European Telecommunication Standards Institute*) sebagian dari pengembangan GSM (*Global System for Mobile Communication*) Phase 2, yang terdapat pada dokumentasi GSM 03.40 dan GSM 03.38. Fitur SMS ini memungkinkan perangkat Stasiun Seluler Digital (*Digital*

Cellular Terminal, seperti Ponsel) untuk dapat mengirim dan menerima pesan-pesan teks dengan panjang sampai dengan 160 karakter melalui jaringan GSM.

Karakter yang dimaksud adalah alphabet A sampai Z, angka 0 sampai 9 dan spasi. Untuk karakter non-Latin, seperti Arab, Kanji atau Mandarin dengan panjang sampai dengan 70 karakter. SMS dapat dikirimkan ke perangkat stasiun seluler digital lainnya hanya dalam beberapa detik selama berada pada jangkauan pelayanan GSM. Lebih dari sekedar pengiriman pesan biasa, layanan SMS memberikan garansi SMS akan sampai pada tujuan meskipun perangkat yang dituju sedang tidak aktif yang dapat disebabkan karena sedang dalam kondisi mati atau berada di luar jangkauan layanan GSM. Dengan adanya feature seperti ini maka layanan SMS juga cocok untuk dikembangkan sebagai aplikasi-aplikasi seperti: pager, e-mail, dan notifikasi voice mail, serta layanan pesan banyak pemakai (multiple user). Namun pengembangan aplikasi tersebut masih bergantung pada tingkat layanan yang disediakan oleh operator jaringan.

2.3.1 Karakteristik SMS

Ada beberapa karakteristik pesan SMS yang penting yaitu :

1. Sebuah pesan singkat yang terdiri dari 160 karakter.
2. Pesan SMS dijamin sampai atau tidak sama sekali selayaknya *e-mail*, sehingga jika terjadi kegagalan sistem atau hal lain yang menyebabkan SMS tidak diterima akan diberikan informasi (*delivery report*) yang menyatakan SMS gagal dikirim.
3. Berbeda dengan fungsi *call* (panggilan), sekalipun saat mengirimkan SMS tetapi handphone tujuan tidak aktif bukan berarti pengiriman SMS akan gagal. Namun SMS akan masuk ke antrian dahulu selama waktu belum time out. SMS akan segera dikirimkan jika handphone sudah aktif.
4. *Bandwidth* yang digunakan rendah.

2.3.2 Keuntungan SMS

Adapun keuntungan dari SMS yaitu:

1. Pengiriman notifikasi dan peringatan (*alert*).
2. Penyampain pesan yang terjamin, handal dan komunikasi dengan biaya rendah.

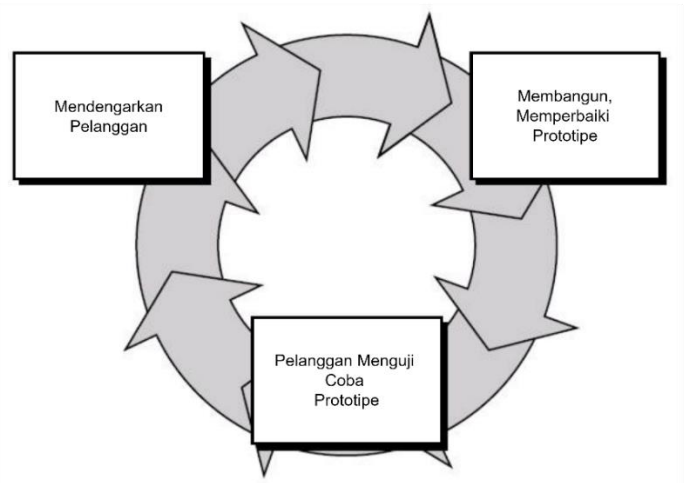
3. Kemampuan untuk menyaring pesan dan menanggapi panggilan secara selektif.
4. Tingkat kegagalan kirim yang sangat kecil sehingga pesan kemungkinan besar akan sampai pada tujuan.

   Pengiriman pesan ke nomor tujuan yang banyak dan berbeda dapat dilakukan pada waktu yang relatif singkat.

2.4 *Prototype*

Menurut (Susanto & Andriana, 2016), Model *prototyping* merupakan suatu teknik untuk mengumpulkan informasi tertentu mengenai kebutuhan-kebutuhan informasi pengguna secara cepat. Berfokus pada penyajian dari aspek-aspek perangkat lunak tersebut yang akan nampak bagi pelanggan atau pemakai. Prototipe tersebut akan dievaluasi oleh pelanggan dan digunakan untuk menyaring kebutuhan pengembangan perangkat lunak.

Menurut (Purnomo, 2017), *prototyping* merupakan metode pengembangan perangkat lunak, yang berupa model fisik kerja sistem dan berfungsi sebagai versi awal dari sistem. Dengan metode *prototyping* ini akan dihasilkan *prototype* dibangun untuk mendifinisikan kebutuhan awal. *Prototype* akan dihilangkan atau ditambahkan pada bagiannya sehingga sesuai dengan perencanaan dan analisis yang dilakukan oleh pengembang sampai dengan uji coba yang dilakukan secara simultan seiring dengan proses pengembangan.



Sumber: Sukamto & Sallahuddin, (2015)

Gambar 2.1 Model *Prototype*

Pengembangan atau perancangan yang akan dilakukan pada penelitian ini menggunakan tiga tahap siklus pengembangan model *prototype* yaitu:

a) Mendengarkan Pelanggan

   Merupakan tahapan pertama dalam merancang sebuah sistem. Pada tahap ini akan menentukan informasi-informasi yang dibutuhkan oleh pelanggan agar tercipta sebuah aplikasi sehingga mengarah pada tujuan dibuatnya aplikasi tersebut.

b) Membangun dan Memperbaiki *Prototype*

   Dalam tahap ini dilakukan perancangan dan pengkodean untuk sistem yang diusulkan yang mana tahapannya meliputi: perancangan proses-proses yang akan terjadi didalam sistem, perancangan diagram UML yang akan digunakan, perancangan antarmuka keluaran serta dilakukan tahap pengkodean terhadap rancangan-rancangan yang telah didefinisikan.

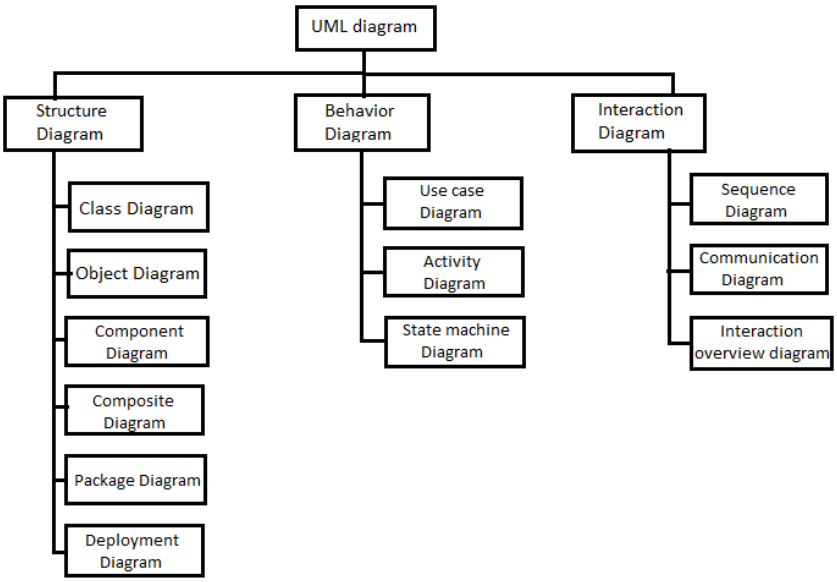
c) Pelanggan Menguji Coba *Prototype*

   Pada tahap ini akan dilakukan pengujian terhadap sistem yang telah disusun dan melakukan pengenalan terhadap sistem yang telah diujikan, serta mengevaluasi apakah sistem yang telah jadi sudah sesuai dengan yang diharapkan.

2.5 *Unified Modeling Language* (UML)

Menurut (Whitten dkk., 2007), *Unified Modeling Language* (UML) merupakan kumpulan dari model yang digunakan untuk menjelaskan sistem dari perangkat lunak sebagai objek. Sedangkan pandangan (Sukamto & Sallahuddin, 2015) UML adalah salah satu standar bahasa visual yang banyak digunakan di dunia industri untuk mengidentifikasi requirement, membuat analisis & desain, serta menggambarkan arsitektur dalam pemrogramman berorientasi objek.

Menurut (Sukamto & Sallahuddin, 2015), dalam UML terdapat 13 diagram yang dikelompokkan dalam 3 kategori. Pembagian kategori dan macam-macam diagram tersebut dapat dilihat pada gambar 2.2 di bawah ini :



Gambar 2.2 Klasifikasi Diagram UML

Berikut penjelasan singkat dari Gambar 2.2 :

1. *Structure Diagram,* yaitu kumpulan diagram-diagram yang menggambarkan struktur statis dari sistem yang dimodelkan.
2. *Behavior Diagram,* yaitu kumpulan diagram-diagram yang menggambarkan kelakuan sistem atau rangkaian perubahan yang terjadi di dalam sistem.
3. *Interaction Diagram,* yaitu kumpulan diagram-diagram yang menggambarkan interaksi sistem dengan sistem lain ataupun interaksi antar subsistem dalam sebuah sistem.

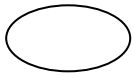
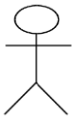
Secara garis besar, beberapa diagram utama sudah dapat menggambarkan keseluruhan sistem. Diagram tersebut antara lain *use case diagram, class diagram, sequence diagram,* dan *activity diagram.*

2.5.1 *Use Case*

Menurut (Sukamto & Sallahuddin, 2015), diagram *use case* merupakan pemodelan untuk kelakuan sistem informasi yang akan dibuat. *Use Case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Secara kasar, *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang

berhak menggunakan fungsi-fungsi itu. Berikut adalah simbol-simbol yang ada pada diagram *use case*.

Tabel 2.1 Simbol Diagram *Use Case*

No.	Notasi	Nama	Deskripsi
1.		Use Case	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor; biasanya dinyatakan dengan menggunakan kata kerja di awal frase nama use case.
2.		Actor	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang, biasanya dinyatakan menggunakan kata benda di awal frase nama aktor.
3.		Association	Komunikasi antara aktor dan use case yang berpartisipasi pada use case atau use case memiliki interaksi dengan aktor.
4.	<<extend>> - - - - - - - ->	Extended (Ekstensi)	Relasi use case tambahan ke sebuah use case di mana use case yang ditambahkan dapat berdiri sendiri walau tanpa use case tambahan itu; mirip dengan prinsip inheritance pada pemrograman berorientasi objek; biasanya use case tambahan memiliki nama depan yang sama dengan use case yang ditambahkan.

No.	Notasi	Nama	Deskripsi
5.	<<include>> < - - - - - - - -	Include (Menggunakan)	Relasi use case tambahan ke sebuah use case di mana use case yang ditambahkan memerlukan use case ini untuk menjalankan fungsinya atau sebagai syarat dijalankan use case ini. Include berarti use case yang ditambahkan akan selalu dipanggil saat use case tambahan dijalankan.

2.5.2 *Class Diagram*

Class diagram menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem (Sukamto & Sallahuddin, 2015). *Class diagram* memiliki apa yang disebut atribut dan operasi. Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas. Operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas. Atribut dan metode dapat memiliki salah satu sifat sebagai berikut:

1. *Private* (-), hanya dapat digunakan oleh *class* yang memilikinya.
2. *Public* (+), dapat digunakan oleh *class* lain.
3. *Protected* (#), hanya dapat dipanggil oleh *class* yang bersangkutan jumlah suatu anak yang mewarisinya.

Nilai kardinalitas atau *multiplicity* sebuah *class* menunjukan jumlah suatu objek yang dapat berhubungan dengan objek lain. Berikut nilai kardinalitas atau *multiplicity* pada Tabel 2.2 (Tohari, 2014) dan notasi *class diagram* pada Tabel 2.3 (Sukamto & Sallahuddin, 2015).

Tabel 2.2 Jenis-jenis *Multiplicity*

No	Indikator	Keterangan
1	0..1	Nol atau satu
2	1	Hanya satu
3	0..*	Nol atau lebih
4	1..*	Satu atau lebih

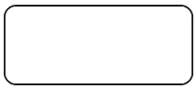
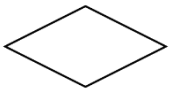
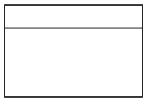
Tabel 2.3 Deskripsi Notasi pada *Class Diagram*

No	Notasi	Nama	Deskripsi
1		*Class*	Menggambarkan konsep dasar pemodelan sistem.
2		Asosiasi (*Association*)	Sebuah garis solid antara dua *class*, ditarik dari *class* sumber ke *class* target lebih spesifik, digunakan dalam struktur pewarisan.
3		Ketergantungan (*Dependency)*	Relasi antara dua elemen jika perubahan definisi sebuah elemen (*supplier* atau sumber) dapat menyebabkan perubahan pada elemen lainnya (*Client* atau target).

2.5.3 *Activity Diagram*

Activity Diagram menggambarkan *workflow* (aliran kerja) proses bisnis dan urutan aktivitas dalam sebuah proses. *Activity diagram* sangat mirip dengan *flowchart* karena memodelkan *workflow* dari satu aktivitas ke aktivitas lainnya atau dari aktivitas ke status. Membuat *activity diagram* pada awal pemodelan proses cukup menguntungkan untuk membantu memahami keseluruhan proses. *Activity diagram* juga bermanfaat untuk menggambarkan *parallel behavior* atau menggambarkan interaksi antara beberapa *use case.* Berikut merupakan simbol notasi *Activity Diagram* pada Tabel 2.1 (Sukamto & Sallahuddin, 2015).

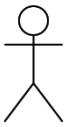
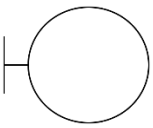
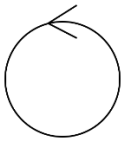
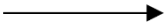
Table 2.1 Simbol *Activity Diagram*

No.	Notasi	Nama	Deskripsi
1		Status Awal (*initial node)*	Status awal aktivitas, sebuah diagram aktivitas memiliki status awal.
2		Status Akhir (*final node)*	Status akhir yang dilakukan sistem.
3		Aktivitas (*activity)*	Aktivitas yang dilakukan oleh sistem, biasanya di awali oleh kata kerja.
4		Percabangan (*decision)*	Asosiasi percabangan di mana jika ada pilihan aktivitas lebih dari satu.
5		Penggabungan (*join)*	Asosiasi penggabungan di mana lebih dari satu aktivitas digabungkan menjadi satu.
6		*Swimlane*	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi.

2.5.4 *Sequence Diagram*

Menurut (Sukamto & Sallahuddin, 2015) berpendapat bahwa, *sequence diagram* menggambarkan alur objek pada *use case* dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek. Oleh karena itu, untuk menggambarkan *sequence diagram* maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu. Berikut adalah simbol-simbol yang ada pada *sequence diagram* pada Tabel 2.4.

Tabel 2.4 Simbol *Sequence Diagram*

No	Notasi	Nama	Deskripsi
1		*Actor*	Menggambarkan orang yang sedang berinteraksi dengan sistem.
2		*Object*	Menyatakan objek yang berinteraksi pesan.
3		*A focus of control & A life line*	Menyatakan kehidupan suatu objek.
4		*Boundary Class*	Menggambarkan sebuah penggambaran dari *form.*
5		*Control Class*	Menggambarkan penghubung antara boundary dengan tabel.
6		*Message*	Menggambarkan pesan atau interaksi antar objek.
7		*Message to self*	Menggambarkan pesan balikan atau reaksi dari objek sebelumnya.

2.6 Teknologi Pendukung

2.6.1 *PHP (Hypertext Preprocessor)*

Menurut (Welling & Thomson, 2009) *PHP* atau *hypertext preprocessor* merupakan bahasa pemrograman *server-side script* yang dirancang untuk pengembangan *web*. Selanjutnya (Arief, 2011) mengungkapkan bahwa *PHP*

merupakan sebuah bahasa *server-side–scripting* yang menyatu dengan *HTML* yang berfungsi untuk membuat halaman *web* yang dinamis. *PHP* merupakan *server-side-scripting* sehingga sintaks dan perintah-perintah dalam *PHP* akan dieksekusi di server, kemudian hasilnya akan dikirimkan ke browser dalam format *HTML*.

Berdasarkan kedua pendapat ahli tersebut, dapat disimpulkan bahwa *PHP* merupakan bahasa yang bersifat *server-side-scripting* yang berguna dalam pembuatan *web* dinamis yang sintaks dan perintahnya dieksekusi di dalam server untuk kemudian dikirimkan hasilnya ke *web browser* dalam bentuk format *HTML*.

2.6.2 *MySQL*

Menurut (Welling & Thomson, 2009), *MySQL* merupakan *DBMS* yang disebarkan secara gratis. *Server MySQL* mengontrol akses ke dalam data agar banyak pengguna bisa mengakses data tersebut secara bersamaan dan memastikan bahwa hanya pengguna tertentu yang dapat mengakses data tersebut. Sedangkan (Arief, 2011), mengungkapkan bahwa *MySQL* adalah salah satu jenis *database server* yang sangat terkenal dan banyak digunakan untuk membangun aplikasi *web* yang menggunakan database sebagai sumber dan pengolahan datanya.

Berdasarkan kedua pendapat ahli tersebut maka dapat disimpulkan bahwa *MySQL* merupakan sebuah *DBMS* yang berfungsi untuk mengontrol akses data kedalam *database* serta melakukan pengelolaan terhadap data yang ada di dalam *database*.

2.6.3 *JavaScript*

Javascript adalah bahasa *scripting* yang populer di internet dan dapat bekerja di sebagian besar browser populer seperti *Internet Explorer* (IE), *Mozilla Firefox, Netscape* dan *Opera*. Kode *Javascript* dapat disisipkan dalam halaman *web* menggunakan tag *SCRIPT* (Sunyoto, 2007). Beberapa hal tentang *Javascript*:

1. *Javascript* didesain untuk menambah interaktif suatu *web*.
2. *Javascript* merupakan sebuah bahasa *scripting*.
3. Bahasa *scripting* merupakan bahasa pemrograman yang ringan.
4. *Javascript* berisi baris kode yang dijalankan di komputer (*web browser*).
5. *Javascript* biasanya disisipkan (*embedded*) dalam halaman HTML.

6. *Javascript* adalah bahasa *interpreter* (yang berarti *script* dieksekusi tanpa proses kompilasi).

2.6.4 *HTML*

HTML (HyperText Markup Language) merupakan salah satu format yang digunakan dalam pembuatan dokumen dan aplikasi yang berjalan di dalam halaman *web* (Arief, 2011). Selanjutnya menurut (Siberio, 2011), *Hyper Text Markup Language atau HTML* adalah bahasa yang digunakan pada dokumen web sebagai bahasa untuk melakukan pertukaran dokumen *web*. Dokumen *HTML* terdiri dari komponen yaitu *tag*, *elemen* dan *attribute*. *Tag* adalah tanda awal < dan tanda akhir > yang digunakan sebagai pengapit suatu elemen. Elemen adalah nama penanda yang diapit oleh tag yang memiliki fungsi dan tujuan tertentu pada dokumen *HTML*. Elemen dapat memiliki elemen anak dan juga nilai. Elemen anak adalah suatu elemen yang berada di dalam elemen pembuka dan elemen penutup induknya. Nilai yang dimaksud adalah suatu teks atau karakter yang berada di antara elemen pembuka dan elemen penutup. Atribut adalah properti elemen yang digunakan untuk mengkhususkan suatu elemen. Elemen dapat memiliki atribut yang berbeda pada tiap masing-masingnya.

Berdasarkan beberapa pendapat ahli tersebut, maka dapat disimpulkan bahwa *HTML* merupakan bahasa yang digunakan dalam pembuatan halaman *web* yang berisi *tag*, *element* dan *attribute* yang memiliki fungsi masing-masing dalam halaman *web*.

2.6.5 *Bootstrap*

Bootstrap merupakan sebuah *framework CSS* yang memudahkan pengembang untuk membangun website yang menarik dan responsif. *Bootstrap* adalah *CSS* tetapi dibentuk dengan *LESS*, sebuah *pre-prosessor* yang memberi fleksibilitas dari penggunaan *CSS* biasa. *Bootstrap* dapat dikembangkan dengan tambahan lainnya karena ini cukup fleksibel terhadap pekerjaan *web* yang mengutamakan desain (Otto, 2011). Selanjutnya (Fauzi, 2008), mengungkapkan bahwa "*Bootstrap* merupakan suatu metode berbasis komputer yang sangat potensial untuk dipergunakan pada masalah ketidakstabilan dan keakurasian, khususnya dalam menentukan interval konfendesi".

Berdasarkan kedua pendapat ahli tersebut, dapat disimpulkan bahwa

bootstrap merupakan sebuah *framework CSS* yang dirancang untuk membantu *developer* dalam melakukan desain terhadap rancangan *website* yang akan dibangun agar dapat lebih menarik, interaktif dan *responsive.*

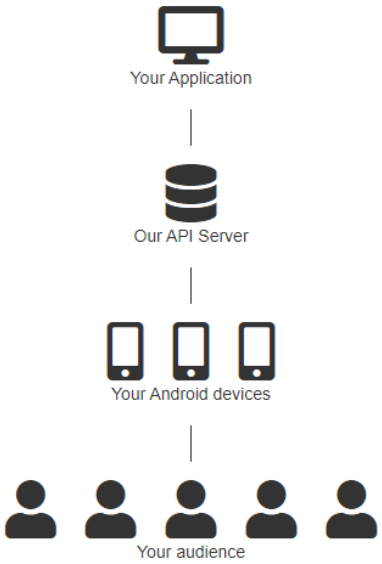
2.6.6 Laravel

Laravel merupakan *framework* PHP yang menekankan pada kesederhanaan dan fleksibilitas pada desainnya. Laravel dirilis di bawah lisensi MIT dengan sumber kode yang disediakan di Github. Sama seperti *framework* PHP lainnya, Laravel dibangun dengan basis MVC (*Model-View-Controller*). Laravel dilengkapi command line tool yang bernama "Artisan" yang bisa digunakan untuk packging bundle dan instalasi bundle. *Framework* Laravel dibuat oleh Taylor Otwell, proyek Laravel dimulai pada April 2011. Awal mula proyek ini dibuat karena Otwell sendiri tidak menemukan *framework* yang up-to-date dengan versi PHP. Mengembangkan *framewrok* yang sudah ada juga bukan merupakan ide yang bagus karena keterbatasan sumber daya. Dikarenakan beberapa keterbatasan tersebut, Otwell membuat sendiri *framework* dengan nama Laravel. Oleh karena itu Laravel mensyaratkan PHP versi 5.3 keatas. (Rohman, 2014).

2.6.7 SMSGateway.me

SMS*Gateway*.me adalah layanan yang memungkinkan kita untuk mengirim dan menerima SMS secara *programmatically* (dari aplikasi) dan menjadikan *Smartphone* Android sebagai perangkatnya (SMS Gateway Me, tanpa tahun).

Syaratnya sederhana, cukup install aplikasi SMS*Gateway*.me di Android, pastikan pulsa untuk mengirim SMS mencukupi, serta *smartphone* dalam keadaan menyala saat digunakan.



Gambar 2.3 Arsitektur SMS*Gateway*.me API

2.7 Pengujian Perangkat Lunak

Pengujian merupakan elemen kritis dari jaminan terhadap kualitas perangkat lunak dan merepresentasikan kajian pokok dari spesifikasi, desain, dan pengkodean. Karakteristik umum dari pengujian perangkat lunak adalah sebagai berikut (Sukamto, 2013) :

1. Pengujian dimulai pada level modul dan bekerja keluar ke arah integrasi pada sistem berbasiskan komputer.
2. Teknik pengujian yang berbeda sesuai dengan poin-poin yang berbeda pada waktunya.
3. Pengujian diadakan oleh *software developer* dan untuk proyek yang besar oleh *group testing* yang *independent*.
4. *Testing* dan *debugging* adalah aktivitas yang berbeda tetapi *debugging* harus diakomodasikan pada setiap strategi *testing.*

Metode pengujian perangkat lunak ada 3 jenis, yaitu (Sukamto, 2013) :

1. *White Box / Glass Box* - pengujian operasi.
2. *Black Box* - untuk menguji sistem.
3. *Use Case* - untuk membuat input dalam perancangan *black box* dan pengujian *statebased.*

Pengujian dalam penelitian ini dilakukan dengan menggunakan metode *Black Box.*

2.7.1 Pengujian *Black Box*

Menurut (Pressman, 2014), *black box testing* juga disebut pengujian tingkah laku, memusat pada kebutuhan fungsional perangkat lunak. Teknik pengujian *black box* memungkinkan memperoleh serangkaian kondisi masukan yang sepenuhnya menggunakan semua persyaratan fungsional untuk suatu program. Selanjutnya (Sukamto & Sallahuddin, 2015) mengungkapkan bahwa *black box testing* merupakan pengujian perangkat lunak dari segi spesifikasi fungsional tanpa menguji desain dan kode program. Pengujian dimaksudkan untuk mengetahui apakah fungsi-fungsi, masukan, dan keluaran dari perangkat lunak sesuai dengan spesifikasi yang dibutuhkan.

Berdasarkan kedua pendapat ahli di atas, dapat disimpulkan bahwa pengujian black box merupakan pengujian perangkat lunak dari segi fungsional suatu program atau aplikasi yang bertujuan untuk mengetahui apakah fungsi-fungsi, masukan, dan keluaran dari program atau aplikasi tersebut telah sesuai dengan spesifikasi yang dibutuhkan.

2.7.2 Pengujian *Skala Likert*

Perhitungan hasil dari kuesioner dilakukan dengan menggunakan cara Skala *Likert.* Skala *Likert* digunakan untuk mengukur sikap, pendapat dan persepsi seseorang atau sekelompok orang tentang fenomena sosial (Sugiyono, 2014). Untuk setiap pilihan jawaban diberi skor, maka responden harus menggambarkan, mendukung pertanyaan dengan jawaban yang dipilih. Dengan skala *likert,* variabel yang akan diukur dijabarkan menjadi indikator variabel. Kemudian indikator tersebut dijadikan sebagai titik tolak ukur menyusun item-item instrumen yang dapat berupa pertanyaan atau pernyataan.

Tabel 2.5 Skala Penilaian Untuk Pertanyaan Positif dan Negatif

Nilai	**Kriteria**
1	Sangat Buruk
2	Buruk
3	Cukup
4	Baik
5	Sangat Baik