



BAB II

TINJAUAN PUSTAKA

2.1 Teori Umum

2.1.1 Pengertian Sistem Informasi

Menurut Kristanto (2018:12), “sebuah sistem informasi merupakan kumpulan dari beberapa perangkat keras dan perangkat lunak komputer serta perangkat manusia yang akan mengolah data menggunakan perangkat keras dan perangkat lunak tersebut”. Hal yang senada pun dikemukakan oleh Sutabri dan Napitupulu (2019:3), “sistem informasi adalah gabungan yang terorganisasi dari manusia, perangkat lunak, perangkat keras, jaringan komunikasi, dan sumber data dalam mengumpulkan, mengubah, dan menyebarkan informasi dalam organisasi”.

Disamping itu, Indrajani menjabarkan lebih rinci sebagaimana dalam bukunya *Database Systems All in One – Theory, Practice, and Case Study* (2018:3), “Sistem informasi merupakan kombinasi teratur apapun dari orang-orang, *hardware*, *software*, jaringan komunikasi, dan sumber daya data yang mengumpulkan, mengubah, dan menyebarkan informasi dalam sebuah organisasi. Sistem informasi juga merupakan sekelompok elemen yang saling berhubungan atau berinteraksi hingga membentuk satu-kesatuan”.

2.1.2 Pengertian Perangkat Lunak (Aplikasi)

Dikutip dari pernyataan Sukamto dan Shalahuddin dalam bukunya yang berjudul *Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Objek* (2013:2), bahwa perangkat lunak (*software*) adalah program komputer yang terasosiasi dengan dokumentasi perangkat lunak seperti dokumentasi kebutuhan, model desain, dan cara penggunaan (*user manual*”). Sedangkan, menurut Sutabri (2014:62), “perangkat lunak aplikasi (*application software*), yaitu program yang ditulis dan diterjemahkan oleh perangkat lunak bahasa untuk menyelesaikan suatu aplikasi tertentu”.



2.1.3 Pengertian Basis Data (*Database*)

Dalam bukunya *Basis Data* revisi kedua, Fathansyah (2015:3) mengemukakan bahwa basis data merupakan kumpulan data yang saling berhubungan dan disimpan secara bersamaan, tanpa disertai pengulangan (redundansi) yang tidak perlu, untuk memenuhi berbagai kebutuhan. Sedangkan, Jubilee Enterprise (2017:1) menyatakan bahwa *database* adalah suatu aplikasi yang menyimpan sekumpulan data, dimana setiap *database* mempunyai perintah tertentu untuk membuat, mengakses, mengatur, mencari, dan menyalin data yang ada didalamnya. Disamping itu, Indrajani (2018:2) menambahkan bahwa basis data adalah kumpulan terpadu dari elemen data logis yang saling berhubungan dan mengonsolidasi banyak catatan yang sebelumnya disimpan dalam file terpisah.

2.2 Teori Khusus

2.2.1 *Unified Modelling Language* (UML)

Menurut Munawar dalam bukunya yang berjudul *Analisis dan Perancangan Sistem Berorientasi Objek dengan UML* (2018:49), “UML (*Unified Modelling Language*) merupakan kesatuan dari bahasa pemodelan yang dikembangkan oleh *Booch*, *Object Modelling Technique* (OMT) dan *Object Oriented Software Engineering* (OOSE), yang 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”.

Disamping itu, Sugiarti (2018:100) mengemukakan bahwa *Unified Modelling Language* (UML) adalah sebuah bahasa standar dalam dunia industri untuk memvisualisasi, merancang dan mendokumentasikan sistem perangkat lunak, dimana UML ini menawarkan sebuah standar dalam perancangan model sebuah sistem.

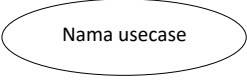




2.2.2 Use Case Diagram

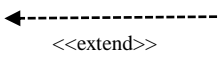
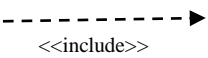
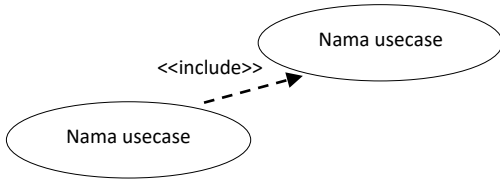
Menurut Sukamto dan Shalahuddin dalam bukunya *Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Objek* (2013:155), “*Use case diagram* atau diagram *use case* merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu 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”.

Disamping itu, menurut Sugiarti (2018:105), “*Use case diagram* secara grafis menggambarkan interaksi antara sistem [...] dan mendeskripsikan siapa yang akan menggunakan sistem dan dalam cara apa pengguna (*user*) mengharapkan interaksi dengan sistem itu”. Adapun simbol atau notasi yang terdapat pada *use case diagram*, diantaranya seperti pada tabel berikut.

Tabel 2.1 Simbol-simbol pada *Use case diagram*

No.	Simbol	Deskripsi
1.	<p><i>Usecase</i></p> 	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar-unit atau aktor; biasanya dinyatakan dengan kata kerja di awal frasa nama <i>usecase</i>
2.	<p>Aktor</p> 	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri.
3.	<p>Asosiasi/<i>association</i></p> 	Komunikasi antara aktor dan <i>usecase</i> yang berpartisipasi pada <i>usecase</i> atau <i>usecase</i> memiliki interaksi dengan aktor.

Lanjutan Tabel 2.1 Simbol-simbol pada *Use case diagram*

No.	Simbol	Deskripsi
4.	<i>Extend</i> 	Relasi <i>usecase</i> tambahan ke sebuah <i>usecase</i> dimana <i>usecase</i> yang ditambahkan dapat berdiri sendiri walau tanpa <i>usecase</i> tambahan itu; Mirip dengan
5.	<i>Include</i> 	Relasi <i>usecase</i> tambahan ke sebuah <i>usecase</i> dimana <i>usecase</i> yang ditambahkan memerlukan <i>usecase</i> ini untuk menjalankan fungsinya atau sebagai syarat dijalankan <i>usecase</i> ini; [...] contoh: 

Sumber: Sugiarti (2018:110-111)




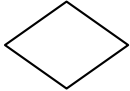


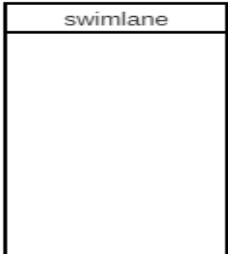
2.2.3 Activity Diagram

Menurut Sukamto dan Shalahuddin (2013:161), “Diagram aktivitas atau *activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau sebuah proses bisnis atau menu yang ada pada perangkat lunak. [...] Diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor”. Disamping itu, Sugiarti (2018:107) menyatakan bahwa terdapat diagram untuk memodelkan *behavior* dari sebuah *usecase* atau sebuah metode, salah satunya ialah *activity diagram* yang secara grafis digunakan untuk menggambarkan rangkaian aliran *activity*, baik dari proses bisnis maupun *usecase*, juga untuk memodelkan



action yang akan dilakukan saat sebuah operasi dieksekusi, dan memodelkan hasil dari *action* tersebut. Berikut merupakan beberapa notasi atau simbol yang terdapat pada *Activity Diagram*.

Tabel 2.2 Simbol-simbol pada *Activity Diagram*

No.	Simbol	Nama	Keterangan
1.		<i>Start state</i>	Titik awal atau permulaan
2.		<i>End state</i>	Titik akhir/akhir dari aktivitas
3.		<i>Activity</i>	Activity atau aktivitas yang dilakukan oleh aktor
4.		<i>Decision</i>	Pilihan untuk mengambil keputusan
5.		<i>Interaction</i>	Alur
6.		Penggabungan/ <i>join</i>	Penggabungan dimana lebih dari satu aktivitas disatukan
7.		Swimlane	Memisahkan entitas yang terdapat pada aktivitas yang terjadi. Dapat digambarkan secara vertikal atau horizontal

Sumber: Rusmawan (2019:80), Sukamto dan Shalahuddin (2013:162)




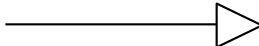
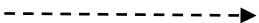

2.2.4 Class Diagram

Menurut Sukamto dan Shalahuddin (2013:141), “Diagram kelas atau *class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem”. Hal yang senada pula dikemukakan Sugiarti (2018:106), “*Class diagram* adalah untuk menggambarkan struktur objek sistem, yang menunjukkan *class object* yang menyusun sistem dan juga hubungan antara *class object* tersebut”. Adapun simbol-simbol yang biasanya digunakan dalam membuat *class diagram*, diantaranya sebagai berikut.

Tabel 2.3 Simbol-simbol pada *Class Diagram*

No.	Simbol	Deskripsi
1.	<p>Operasi/kelas</p>	<i>Class</i> pada struktur sistem
2.	<p>Antarmuka/<i>interface</i></p>	Sama dengan konsep <i>interface</i> dalam pemrograman berorientasi objek
3.	<p>Asosiasi</p>	Relasi antar <i>class</i> dengan makna <i>class</i> yang satu digunakan oleh <i>class</i> yang lain. Asosiasi biasanya juga disertai dengan <i>multiplicity</i> (jumlah suatu objek yang bisa berhubungan dengan objek lain, seperti “satu” atau “banyak”).
4.	<p>Asosiasi berarah/<i>directed association</i></p>	Relasi antar- <i>class</i> disertai biasanya juga disertai dengan <i>multiplicity</i>

Lanjutan **Tabel 2.3** Simbol-simbol pada *Class Diagram*

No.	Simbol	Deskripsi
5.	Asosiasi berarah/ <i>directed association</i> 	Relasi antar- <i>class</i> dengan makna <i>class</i> yang satu digunakan oleh <i>class</i> yang lain, asosiasi biasanya juga disertai dengan <i>multiplicity</i>
6.	Generalisasi 	Relasi antar- <i>class</i> dengan makna generalisasi spesialisasi (umum khusus)
7.	Kebergantungan/ <i>dependency</i> 	Relasi antar- <i>class</i> dengan makna kebergantungan antar- <i>class</i>
8.	Agregasi 	Relasi antar- <i>class</i> dengan makna semua-bagian (<i>whole-part</i>)

Sumber: Sugiarti (2018:122)


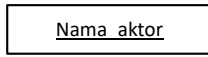
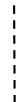
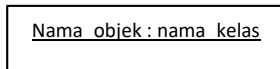
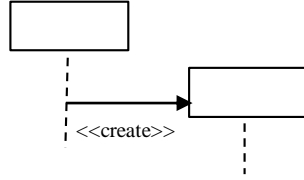

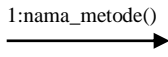
2.2.5 Sequence Diagram

Menurut Sukanto dan Shalahuddin (2013:165), “Diagram sekuen menggambarkan kelakuan objek pada *usecase* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek. Oleh karena itu, untuk menggambar diagram sekuen maka harus diketahui objek-objek yang terlibat dalam sebuah *usecase* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu”.

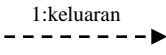
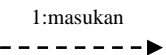
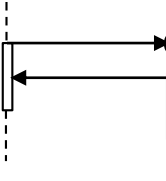
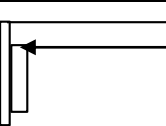
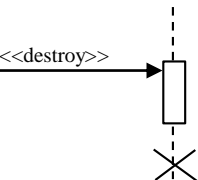
Disamping itu, Sugiarti (2018:106) mengungkapkan lebih rinci bahwa secara grafis, *sequence diagram* (diagram rangkaian) adalah suatu diagram yang menggambarkan bagaimana suatu objek berinteraksi dengan satu sama lain melalui pesan pada sekuensi sebuah *usecase* atau operasi, serta dapat mengilustrasikan bagaimana pesan terkirim dan diterima antara objek dalam suatu *sequence* atau *timing*.

Ada beberapa simbol yang biasanya dipergunakan dalam *sequence diagram*, diantaranya sebagai berikut.

Tabel 2.4 Simbol-simbol pada *Sequence Diagram*

No.	Simbol	Nama	Keterangan
1.	 atau  tanpa waktu aktif	<i>Actor</i>	Orang ataupun pihak yang akan mengelola sistem
2.		<i>Lifeline</i>	Menggambarkan sebuah objek dalam sebuah sistem atau salah satu komponennya
3.		Objek	Menyatakan objek yang berinteraksi pada pesan
4.		<i>Create message</i>	Pembuatan sebuah message sederhana antarelemen dan juga mengindikasikan komunikasi antara objek
5.		Waktu aktif	Menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan didalamnya. Aktor tidak memiliki waktu aktif.
6.		Pesan tipe <i>call</i>	Menyatakan suatu objek memanggil suatu operasi/metode yang ada pada objek lain atau dirinya sendiri

Lanjutan Tabel 2.4 Simbol-simbol pada *Sequence Diagram*

No.	Simbol	Nama	Keterangan
1.		Pesan tipe <i>return</i>	Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode, menghasilkan suatu kembalian ke objek tertentu. Arah panah mengarah pada objek yang menerima kembalian.
2.		Pesan tipe <i>send</i>	Menyatakan bahwa suatu objek mengirimkan data/input/informasi ke objek lainnya.
3.		<i>Synchronous message</i>	<i>Message</i> ini mengaktifkan sebuah proses dan sampai selesai, baru bisa mengirimkan sebuah <i>message</i> baru
4.		<i>Message to self (call)</i>	Suatu hasil kembalian sebuah operasi dan berjalan pada objek itu sendiri
5.		Pesan tipe <i>destroy</i>	Menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri, sebaiknya jika ada <i>create</i> maka ada <i>destroy</i>

Sumber: Rusmawan (2019:84-85), Sukanto dan Shalahuddin (2013:165-167)



2.2.6 Black-Box Testing

Menurut Rahayu (dalam Rusmawan, 2019:112), “*Black-box testing* adalah tipe *testing* yang memperlakukan perangkat lunak yang tidak diketahui kinerja internalnya. Sehingga, para *tester* memandang perangkat lunak seperti layaknya sebuah “kotak hitam” yang tidak penting dilihat isinya, tapi cukup dikenai proses *testing* di bagian luar”. Hal serupa diungkapkan Sukamto dan Shalahuddin (2013:275), “*Black-box testing* (pengujian kotak hitam) yaitu menguji 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”.

Sukamto dan Shalahuddin (2019:275-276) kemudian menambahkan, bahwa pengujian ini dilakukan dengan membuat kasus uji yang bersifat mencoba semua fungsi dengan memakai perangkat lunak, apakah sesuai dengan spesifikasi yang dibutuhkan atau tidak. Baik untuk kasus yang benar maupun salah. Selanjutnya, menurut Rusmawan (2019:112), ada beberapa keuntungan yang dapat diperoleh dari jenis *testing* ini, antara lain:

1. Anggota tim *tester* tidak harus seseorang yang memiliki kemampuan teknis di bidang pemrograman;
2. Kesalahan dari perangkat lunak ataupun seringkali ditemukan oleh komponen *tester* yang berasal dari pengguna;
3. Hasil dari *testing* ini dapat memperjelas kontradiksi ataupun kerancuan yang mungkin ditimbulkan dari eksekusi perangkat lunak;
4. Proses *testing* dapat dilakukan lebih cepat dibandingkan *white box testing*.

2.3 Pengertian Judul

2.3.1 Pengertian Integrasi

Menurut Afandi, et. al. (2016:4), “[...] integrasi didefinisikan sebagai adanya saling keterkaitan antara subsistem sehingga data dari satu sistem secara rutin dapat melintas menuju, atau diambil oleh satu atau lebih sistem



yang lain”. Sedangkan, pengertian integrasi dalam konteks Teknologi Informasi (TI) menurut Pratama (2016:20), “Integrasi (*Integration*) didefinisikan sebagai sebuah proses yang meliputi perakitan sejumlah komponen atau bagian-bagian logik dari beberapa buah sistem untuk menjadi sebuah kesatuan sistem, pengecekan jalannya sistem dan fungsionalitas sistem, yang dilakukan sebagai upaya untuk penghematan anggaran IT”.

2.3.2 Sekilas Mengenai Sistem Informasi Geografis (SIG)

Menurut Riyanto (dalam Ariyanti, et. al., 2015:120), “Sistem Informasi Geografis (SIG) adalah sistem informasi khusus yang mengelola data yang memiliki informasi spasial (bereferensi keruangan), atau dalam arti yang lebih sempit adalah sistem komputer yang memiliki kemampuan untuk membangun, menyimpan, mengelola, dan menampilkan informasi bereferensi geografis [...]”. Disamping itu, pendapat yang serupa disampaikan Adelino (2015:17), “SIG (Sistem Informasi Geografis) adalah suatu sistem yang menyajikan gambar, mengecek, mengintegrasikan, memanipulasi, menganalisa, dan menampilkan data yang secara spasial (keruangan) mereferensikan kepada kondisi bumi”.

2.3.3 Pengertian *Mobile Web*

Menurut Utomo dalam bukunya yang berjudul *Mobile Web Programming* (2013:3), “*Web mobile* merupakan *web* atau halaman *website* internet yang dapat digunakan atau diakses pada perangkat *mobile*”. Serupa dengan Usmanto, et. al. (2018:34), “[...] *website mobile* adalah situs yang dirancang khusus untuk perangkat *mobile* yang dirancang menggunakan standar dan protokol yang sama dengan *desktop web*”.

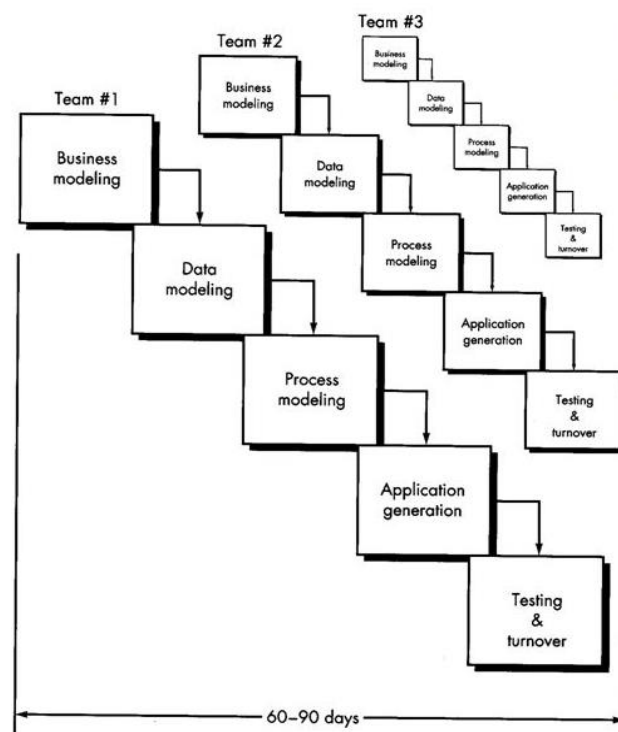
2.3.4 Model *Rapid Application Development* (RAD)

Menurut Vargas-Pérez, et. al., (2016:50), “*Rapid Application Development (RAD)* is a method used for the elaboration of software products



mainly based upon the continous interactive prototyping and design of the system with a huge involvement and participation of final users by means of computerized tools”. Dimana, dalam jurnalnya tersebut dikatakan bahwa model *Rapid Application Development* (RAD) merupakan metode penjabaran produk perangkat lunak yang terutama didasarkan pada desain *prototyping* interaktif berkelanjutan dan desain sistem dengan keterlibatan besar dan partisipasi pengguna akhir melalui komputerisasi.

Sedangkan, menurut Sukamto (2013:34), “*Rapid Application Development* (RAD) adalah model proses pengembangan perangkat lunak yang bersifat inkremental terutama untuk waktu pengerjaan yang pendek. Model RAD adalah adaptasi model air terjun (*waterfall*) versi kecepatan tinggi dengan menggunakan model air terjun untuk pengembangan setiap komponen perangkat lunak”. Adapun ilustrasi model *Rapid Application Development* (RAD) menurut Sukamto dan Shalahuddin (2013:35-36), seperti gambar berikut ini.



Gambar 2.1 Model *Rapid Application Development* (RAD)



Keterangan:

1. Pemodelan bisnis

Pemodelan yang dilakukan untuk memodelkan fungsi bisnis untuk mengetahui informasi apa yang terkait proses bisnis, informasi apa saja yang harus dibuat, siapa yang harus membuat informasi, bagaimana alur informasi itu, proses apa saja yang terkait informasi itu.

2. Pemodelan data

Memodelkan data apa saja yang dibutuhkan berdasarkan pemodelan bisnis dan mendefinisikan atribut-atributnya beserta relasinya dengan data-data yang lain.

3. Pemodelan proses

Mengimplementasikan fungsi bisnis yang sudah didefinisikan terkait dengan pendefinisian data.

4. Pembuatan aplikasi

Mengimplementasikan pemodelan proses dan data menjadi program. Model RAD sangat menganjurkan pemakaian komponen yang sudah ada jika dimungkinkan.

5. Pengujian dan pergantian

Menguji komponen-komponen yang dibuat. Jika sudah teruji maka tim pengembang komponen dapat beranjak untuk mengembangkan komponen berikut

2.3.5 Pengertian dari “Integrasi Sistem Informasi Geografis (SIG) pada Aplikasi *Delivery Point* Berbasis *Mobile Web* dengan Implementasi Model *Rapid Application Development* (RAD) pada Kantor Regional III Palembang PT Pos Indonesia (Persero)”

Integrasi Sistem Informasi Geografis pada Aplikasi *Delivery Point* Berbasis *Mobile Web* dengan Implementasi Model RAD pada instansi tersebut bermakna sebagai suatu proses pengaitan subsistem dari komponen sistem informasi bereferensi keruangan (geografis) untuk menjadi suatu



kesatuan dengan aplikasi *delivery point* (titik pengiriman) Kantor Regional III Palembang PT Pos Indonesia (Persero) dengan platform mobile web. Dimana dalam prosesnya, menerapkan metode pengembangan perangkat lunak *Rapid Application Development* (RAD).

2.4 Teori Program

2.4.1 Sekilas Mengenai MySQL



Gambar 2.2 Logo MySQL

Menurut Murya (2014:46), “MySQL adalah sebuah perangkat lunak sistem manajemen basis data SQL (*Database Management System*) atau DBMS yang *multithread* dan *multi-user*”. Sedangkan, menurut Hendry (2015:7), “MySQL adalah sebuah implementasi dari sistem manajemen basis data relasional yang didistribusikan secara gratis di bawah lisensi GPL (*General Public License*). [...] MySQL sebenarnya merupakan turunan salah satu konsep utama dalam basis data yang telah ada sebelumnya SQL (*Structured Query Language*)”.

2.4.2 Pengertian Google Maps API

Menurut Rismayani (2016:188-189), “Google Maps API merupakan suatu fitur aplikasi yang dikeluarkan oleh google untuk memfasilitasi pengguna yang ingin mengintegrasikan Google Maps ke dalam *website* masing-masing dengan menampilkan data *point* milik sendiri”. Sedangkan menurut Mahdia, et. al. (2013:164), “Google Maps API adalah sebuah layanan (*service*) yang diberikan Google kepada para pengguna untuk memanfaatkan Google Maps dalam mengembangkan aplikasi”.



Disamping itu, menurut Purnomo dalam jurnalnya (2015:2), “Google Maps API merupakan fasilitas dari Google yang dapat digunakan untuk menambahkan peta kedalam *website* dengan menggunakan JavaScript. Google Maps API menyediakan banyak fasilitas dan utilitas untuk memanipulasi peta dan menambahkan konten ke peta melalui berbagai layanan”.

2.4.3 Pengertian PHP



Gambar 2.3 Logo PHP

Menurut Raharjo, dkk pada bukunya yang berjudul *Modul Pemrograman Web (HTML, PHP, dan MySQL)* edisi revisi ke-2 (2014:47), “PHP adalah salah satu bahasa pemrograman skrip yang dirancang untuk membangun aplikasi *web*”. Disamping itu, menurut Sidik (2014:4), “PHP merupakan secara umum dikenal sebagai bahasa pemrograman *script* yang membuat dokumen HTML secara *on the fly* yang dieksekusi di *server web* [...]”.

2.4.4 Pengertian Netbeans

Menurut Sugiarti (2018:6), “Netbeans merupakan IDE yang ditujukan untuk memudahkan pemrograman Java. Dalam Netbeans, pemrograman dilakukan berbasis visual”.



Gambar 2.4 Logo Netbeans



Disamping itu, menurut Dhika, et. al. (2019:105), “NetBeans yaitu lingkungan pengembangan yang bebas, *open source*, terintegrasi (IDE) yang memungkinkan untuk mengembangkan aplikasi *desktop*, seluler dan *web*. IDE ini mendukung pengembangan aplikasi dalam berbagai bahasa, seperti Java, HTML5, PHP, C++. IDE ini menyediakan dukungan terintegrasi untuk siklus pengembangan lengkap, dari penciptaan proyek melalui *debugging*, profil, dan penyebaran. IDE ini berjalan pada Windows, Linux, Mac OS X, dan sistem berbasis UNIX lainnya”.