BAB II
LANDASAN TEORI

2.1. Tinjauan Pustaka

Berikut ini diantara penelitian sebelumnya yang dapat penulis dokumentasikan sebagai tinjauan pustaka:

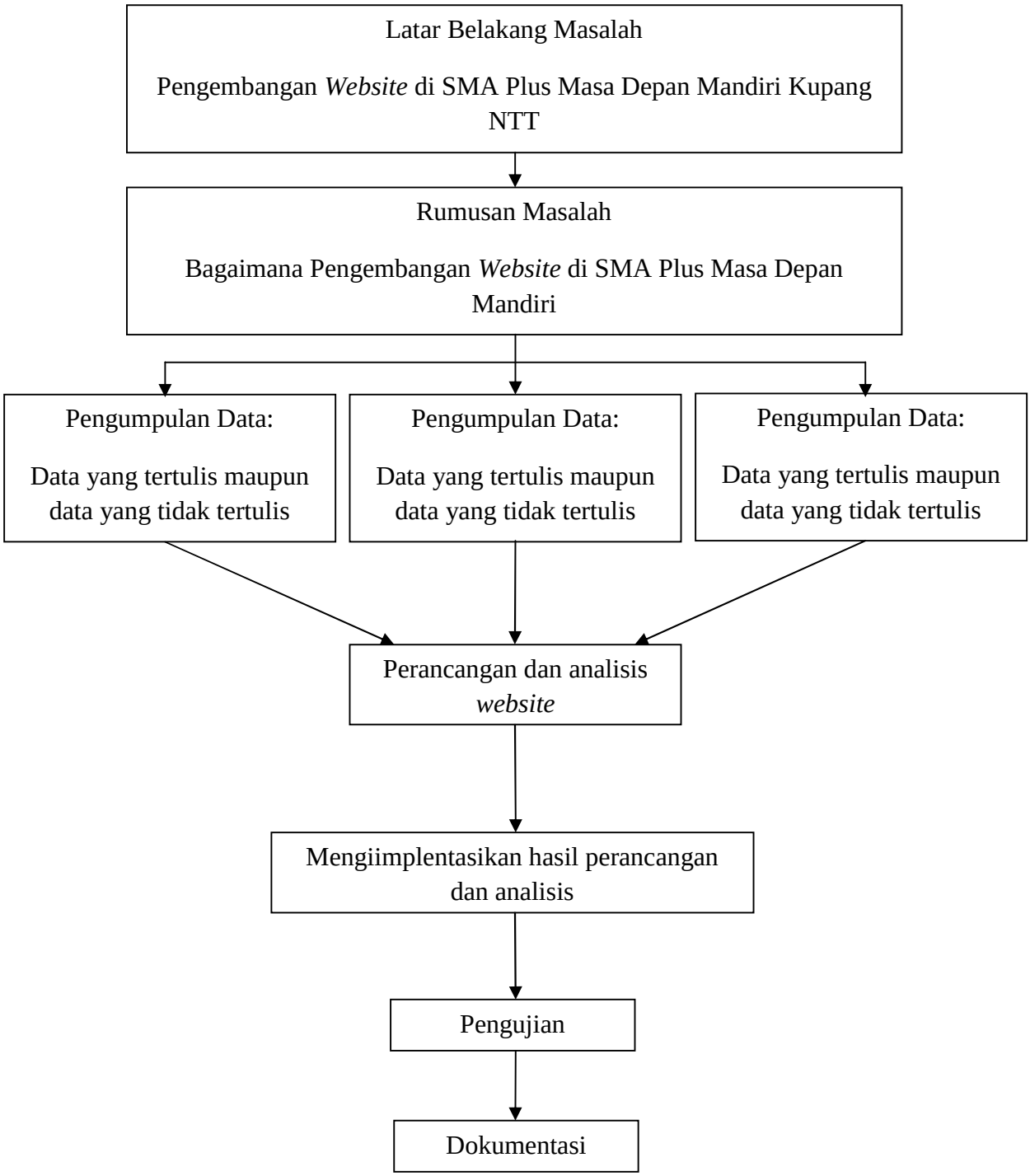
Penelitian yang telah dilakukan oleh Winoto dan Irianto (2014) Mahasiswa Teknik Informatika Universitas Surakarta Jawa Tengah yang mengambil tema mengenai situs *web* profil sekolah yang diimplemtasikan di Sekolah Dasar Negeri 03 Kalisoro, sistem ini mengharapkan untuk menyampaikan dan mengenalkan SD N 03 Kalisoro agar lebih dikenal masyarakat.

Penelitian yang dilakukan oleh Soleh (2014) Pelajar Sistem Informasi Universitas Widiatama Bandung yang mengambil tema tentang situs *web* profil Sekolah yang diimplemntasikan di SMK Islam Sudirman Kedungjati. Sistem ini mengharapkan untuk menyimpan informasi dan mengenalkan SMK Islam Sudirman Kedungjati agar dikenal masyarakat luas dan meningkatkan jumlah pendaftar siswa dan siswi SMK.

Penelitian yang telah dilakukan oleh Nurhadmadi (2014) Mahasiswa Teknik Informatika STIMIK AMIKOM Yogyakarta yang mengambil tema mengenai situs *web* profil sekolah dan *e-learning* yang diimplemntasikan di SMP N 7 Yogyakarta, sistem informasi ini mengharapkan untuk dengan mudah unggah dan unduh pelajaran dan tugas sekolah.

2.2. Kerangka Pemikiran

Kerangka pemikiran adalah suatu diagram yang menjelaskan secara garis besar alur logika berjalannya sebuah penelitian. Kerangka pemikiran dibuat berdasarkan pertanyaan penelitian (*research question*) dan mempresentasikan suatu himpunan dari beberapa konsep serta hubungan diantara konsep-konsep tersebut (Polancik, 2015).



Gambar 1. Diagram Kerangka Pemikiran

2.3. Teori Pendukung

2.3.1. Pengertian *Website*

Website merupakan kumpulan halaman-halaman yang digunakan untuk menampilkan informasi teks, gambar diam atau gerak, animasi, suara, dan atau gabungan dari semuanya, baik yang bersifat statis maupun dinamis yang membentuk satu rangkaian bangunan yang saling terkait, yang masing- masing masing dihubungkan dengan jaringan-jaringan halaman (Bekti, 2015).

2.3.2. Pengertian *Web Server*

Web Server adalah software yang memberikan layanan data yang mempunyai fungsi untuk menerima permintaan HTTP (*Hypertext Transfer Protocol)* atau HTTPS yang dikirim oleh klien melalui *web browser* dan mengirimkan kembali hasilnya dalam bentuk halaman *web* yang umumnya berbentuk HTML (*Hypertext Markup Language). Web server* berguna sebagai tempat aplikasi *web* dan sebagai penerima *request dari client (*Indra Warman, 2014).

2.4. Metode Pengembangan Sistem

SDLC atau lebih dikenal *System Development Life Cycle* adalah proses mengembangkan atau mengubah suatu sistem perangkat lunak dengan menggunakan model-model dan metodeologi yang digunakan orang untuk mengembangkan sistem-sistem perangkat sebelumnya berdasarkan *best practice* atau cara-cara yang sudah teruji baik (Rosa dan M. Shalahuddin, 2014).

2.5. Analisis Sistem

2.5.1. Pengertian *Unifed Modeling Language* (UML)

Berikut ini definisi UML menurut para ahli :

1) Menurut Nugroho (2014) “UML (Unified Modeling Language) adalah ‘bahasa’ pemodelan untuk sistem atau perangkat lunak yang berparadigma ‘berorientasi objek”. Pemodelan (modeling) sesungguhnya digunakan untuk penyederhanaan permasalahan-permasalahan yang kompleks sedemikian rupa sehingga lebih mudah dipelajari dan dipahami”.

Berdasarkan beberapa pendapat yang dikemukakan diatas dapat ditarik kesimpulan bahwa “UML adalah sebuah bahasa yang berdasarkan grafik atau gambar untuk memvisualisasikan, menspesifikasikan, membangun dan pendokumentasian dari sebuah system pengembangan perangkat lunak berbasis OO (*Object Oriented*)”.

Langkah-langkah Penggunaan UML Menurut Sukamto dan Salahuddin (2014). Langkah-langkah penggunaan UML sebagai berikut:

1) Buatlah daftar *business process* dari level tertinggi untuk mendefinisikan aktivitas dan proses yang mungkin muncul.
2) Petakan *use case* untuk setiap *business process* untuk mendefinisikan dengan tepat fungsional yang harus disediakan oleh sistem, kemudian perhalus *use case diagram* dan lengkapi dengan *requirement, constraints* dan catatan-catatan lain.
3) Buatlah *deployment diagram* secara kasar untuk mendefinisikan arsitektur fisik sistem.
4) Definisikan *requirement* lain non fungsional, *security* dan sebagainya yang juga harus disediakan oleh sistem.
5) Berdasarkan *use case diagram*, mulailah membuat *activity diagram*.
6) Definisikan obyek-obyek level atas *package* atau domain dan buatlah *sequence* dan/atau *collaboration* utuk tiap alir pekerjaan, jika sebuah *use case* memiliki kemungkinan alir normal dan *error*, buat lagi satu diagram untuk masing-masing alir.
7) Buatlah rancangan *user interface* model yang menyediakan antamuka bagi pengguna untuk menjalankan skenario *use case.*
8) Berdasarkan model-model yang sudah ada, buatlah *class diagram*. Setiap *package* atau domain dipecah menjadi *hirarki class* lengkap dengan atribut dan metodenya. Akan lebih baik jika untuk setiap *class* dibuat unit *test* untuk menguji fungsionalitas *class* dan interaksi dengan *class* lain.
9) Setelah *class* diagram dibuat, kita dapat melihat kemungkinan pengelompokkan *class* menjadi komponen-komponen karena itu

buatlah *component diagram* pada tahap ini. Juga, definisikan *test* integrasi untuk setiap komponen meyakinkan ia bereaksi dengan baik.

10) Perhalus *deployment diagram* yang sudah dibuat. Detilkan kemampuan dan *requirement* piranti lunak, sistem operasi, jaringan dan sebagainya. Petakan komponen ke dalam node.
11) Mulailah membangun sistem. Ada dua pendekatan yang tepat digunakan :

    a) Pendekatan *use case* dengan mengassign setiap *use case* kepada tim pengembang tertentu untuk mengembangkan unit kode yang lengkap dengan test.

    b) Pendekatan komponen yaitu mengassign setiap komponen kepada tim pengembang tertentu.
12) Lakukan uji modul dan uji integrasi serta perbaiki model beserta kodenya. Model harus selalu sesuai dengan kode yang aktual.
13) Perangkat lunak siap diimplemtasikan".

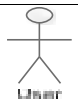
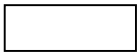
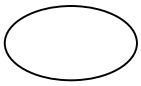
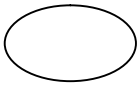
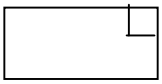
2.5.2. Macam UML

Macam diagram dalam UML, yaitu :

1) *Use Case Diagram*

Diagram ini memperlihatkan himpunan *use case* dan actor (suatu jenis khusus dari kelas). Diagram ini terutama sangat penting untuk mengorganisir dan memodelkan perilaku dari suatu system yang dibutuhkan serta diharapkan pengguna. Simbol *Use Case Diagram* seperti yang tertera pada Tabel 2.1.

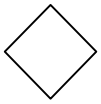
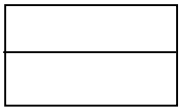
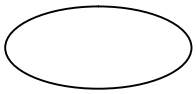
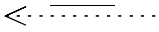
Tabel 2.1 Simbol *Use Case Diagram*

NO	GAMBAR	NAMA	KETERANGAN
1		*Actor*	Menspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan *use case*
2		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (*independent*) akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri (*independent*)
3		*Generalization*	Hubungan dimana objek anak (Descendent) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*anc*estor).
4		*Include*	Menspesifikasikan bahwa *use case* sumber daya *ekspisit*
5		*Extend*	Menspesifikasikan bahwa use case target memperluas perilaku dari *use case* sumber pada satu titik yang diberikan
6		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya.
7		*System*	Menspesifikasikan paket yang menampilkan sistem secara terbatas
8		*Use Case*	Deskripsi dari urutan aksi-aksi yang ditampilkan system yang menghasilkan suatu hasil yang terukur bagi suatu actor
9		*Collaboration*	Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan perilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi)
10		Note	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya

2) *Class Diagram*

Diagram ini memperlihatkan himpunan kelas-kelas, antarmuka-antarmuka, kolaborasi-kolaborasi, relasi-relasi antar objek. Simbol *Class Diagram* seperti yang tertera dalam Tabel 2.2.

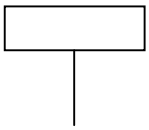
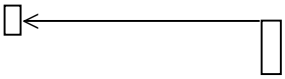
Tabel 2.2 Simbol *Class Diagram*

NO	GAMBAR	NAMA	KETERANGAN
1		*Generalization*	Hubungan dimana objek anak (Descendent) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*anc*estor).
2		*Nary Association*	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek
3		*Class*	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama
4		*Collaboration*	Deskripsi dari urutan-urutan yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu actor
5		*Realization*	Operasi yang benar-benar dilakukan oleh suatu objek
6		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (*independent*) akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri (*independent*)
7		*Association*	Hubungkan antara objek satu dengan objek lainnya.

3) *Sequnce Diagram*

Diagram ini memperlihatkan interaksi yang menekankan pada pengiriman pesan (*message*) dalam suatu waktu tertentu. Simbol *Sequence Diagram* Seperti yang tertera dalam Tabel 2.3.

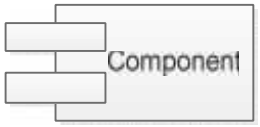
Tabel 2.3 Simbol *Sequence Diagram*

NO	GAMBAR	NAMA	KETERANGAN
1		*LifeLine*	Objek entity, antarmuka yang saling berinteraksi
2		*Message*	Spesifikasi dari komunikasi antar objek yang membuat informasi-informasi tentang aktifitas yang terjadi
3		*Message*	Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktifitas yang terjadi

4) *Component Diagram*

Diagram ini untuk menunjukan organisasi dan ketergantungan diantara kumpulan komponen dalam sebuah sistem. Simbol *Component Diagram* seperti yang tertera dalam Tabel 2.4.

Tabel 2.4 Simbol *Component Diagram*

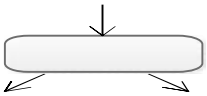
NO	SIMBOL	NAMA	DESKRIPSI
1		*Package*	Package merupakan sebuah bungkusan dari satu atau lebih komponen
2		Komponen	Komponen system
3		Ketergantungan (*Dependency*)	Kebergantungan antar komponen, arah panah mengarah pada komponen yang dipakai
4		Antarmuka (*Interface*)	Sama dengan interface pada pemograman berbasis objek, yaitu sebagai antarmuka komponen tidak mengakses langsung komponen
5		*Link*	Relasi antar komponen

5) *Activity Diagram*

Diagram ini memperlihatkan aliran dari suatu aktifitas ke aktifitas lainnya dalam suatu system. Diagram ini terutama penting dalam pemodelan

fungsi-fungsi dalam suatu system dan member tekanan pada aliran kendali antar objek. Simbol *Activity Diagram* seperti yang tertera dalam Tabel 2.5

Tabel 2.5 Simbol *Activity Diagram*

NO	GAMBAR	NAMA	KETERANGAN
1		*Activity*	Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain
2		*Action*	State dari sistem yang mencerminkan eksekusi dari suatu aksi
3		*Initial Node*	Bagaimana objek dibentuk atau diawali
4		*Actifity Final Node*	Bagaimana objek dibentuk dan dihancurkan
5		*Fork Nod*e	Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran

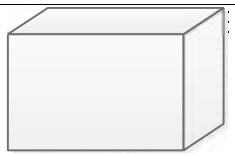
6) *Deployment Diagram*

Deployment diagram merupakan gambaran proses-proses berbeda pada suatu sistem yang berjalan dan bagaimana relasi di dalamnya. Hal inilah yang mempermudah user dalam pemakaian sistem yang telah dibuat dan diagram tersebut merupakan diagram yang statis. Misalnya untuk mendeskripsikan sebuah situs web, deployment diagram menunjukkan komponen perangkat keras ("node") apa yang digunakan (misalnya, web server, server aplikasi, dan database server), komponen perangkat lunak ("artefak") apa yang berjalan pada setiap node (misalnya, aplikasi web,

database), dan bagaimana bagian-bagian yang berbeda terhubung (misalnya JDBC, REST, RMI).

Node digambarkan sebagai kotak, dan artefak yang dialokasikan ke setiap node digambarkan sebagai persegi panjang di dalam kotak. Node mungkin memiliki subnodes, yang digambarkan sebagai kotak nested. Sebuah node tunggal secara konseptual dapat mewakili banyak node fisik, seperti sekelompok database server. Simbol *Deployment Diagram* seperti yang tertera dalam Tabel 2.6.

Tabel 2.6 Simbol *Deployment Diagram*

NO	SIMBOL	NAMA	KETERANGAN
1		*Compon*ent	*Pada D*eployment diagram, komponen yang ada diletakan didalam node untuk memastikan keberadaan posisi mereka
2		*Node*	*Nod*e menggambarkan bagian hardware dalam sebuah sistem. Notasi untuk node yang digambarkan sebagai sebuah kubus 3 dimensi
3		*Association*	Sebuah association dinggambarkan sebagai sebuah garis yang menghubungkan dua node yang mengindikasikan jalur komunikasi antara komponen-komponen hardware
4		*Dependecy*	Kebergantungan antar node, arah panah mengarah pada node yang dipakai

2.6. Software Pendukung

2.6.1. HTML

HTML (*Hypertext Markup Language*) adalah bahasa standar untuk membuat halaman-halaman *web*, sedangkan PHP (*PHP Hypertext preprocessor*) berkedudukan sebagai tag dalam bahasa HTML. Model kerja HTML diawali dengan permintaan suatu halaman *web* oleh *browser,* dari *browser* permintaan dilanjutkan ke *webserver* yang kemudian mencarikan file yang diminta dan memberikan isinya ke *browser.* Perbedaanya jika menggunakan kode atau tag PHP adalah ketika berkas PHP yang diminta oleh *browser* didapatkan oleh *webserver,* isinya segera dikirimkan ke mesin PHP dan mesin inilah yang memproses dan memberikan hasilnya (berupa kode HTML) yang kemudian akan dikirim ke *browser* oleh *webserver*. Secara khusus, PHP dirancang untuk membentuk aplikasi *web* dinamis (Bekti, 2015).

2.6.2. *XAMPP Control Panel*

XAMPP adalah sebuah *software* yang berfungsi untuk menjalankan *website* berbasis PHP dan menggunakan pengolah data MYSQL di komputer lokal". XAMPP berperan sebagai *server web* pada komputer lokal. XAMPP juga dapat disebut sebuah *Cpanel server virtual*, yang dapat membantu melakukan *preview* sehingga dapat dimodifikasi website tanpa harus *online* atau terakses dengan *internet* menurut (Madcoms, 2016:148).

2.6.3. MySQL

MySQL adalah salah satu *database management system* (DBMS) dari sekian banyak DBMS seperti *Oracle*, MS SQL, *Postagre* SQL dan lainya. MySQL berfungsi untuk mengolah *database* menggunakan bahasa SQL. MySQL bersifat *open source* sehingga bisa digunakan secara gratis. Pemograman PHP juga sangat mendukung dengan *database* MySQL (Madcoms, 2016:17).

2.6.4. PHP

Desain PHP singkatan dari *Hypertext Preprocessor* yaitu bahasa pemograman *web server-side* yang bersifat *open source.* PHP merupakan script yang terintegrasi dengan HTML yang berada pada *server* (*server side HTML*

embedded scripting). PHP adalah script yang digunakan untuk membuat halaman yang dinamis atau yang *up to date* (Madcoms, 2016;148).

Kode PHP mempunyai beberapa ciri khusus, yaitu :

1) Hanya dapat dijalankan menggunakan web server, misalnya; *Apache*.
2) Kode PHP diletakkan dan dijalankan di *web server*.
3) Kode PHP dapat digunakan untuk mengakses database, seperti MySQL, *Postgre*SQL, *Oracle*, dan lain – lain.
4) Merupakan s*oftware* yang bersifat *open source*.
5) Gratis untuk di-*download* dan digunakan.
6) Memiliki sifat *multiplatform*, artinya dapat dijalankan menggunakan sistem operasi apapun, seperti: *Linux, Unix, Windows,* dan lain – lain.
7) UML *Unifed Modeling Language* (UML) adalah keluaran notasi grafis yang didukung oleh.

2.6.5. *Adobe Photoshop*

Adobe Photoshop adalah perangkat lunak *editor* buatan *Adobe System* yang dikhususkan untuk pengeditan foto atau gambar dan pembuatan efek. Perangkat ini banyak digunakan oleh photografer digital dan perusahaan iklan sehingga dianggap sebagai pemimpin pasar (*Market Leader*) untuk perangkat lunak pengolahan gambar atau foto dan bersama *Adobe Acrobat* dianggap sebagai produk terbaik yang pernah diproduksi oleh *Adobe System* (Sadeli, 2014:12).