BAB II

LANDASAN TEORI

2.1 Penelitian Terdahulu

Hasil penelitian terdahulu dalam penelitian ini digunakan sebagai salah satu dasar untuk mendapatkan gambaran dalam meyusun kerangka berfikir penelitian, dan menjadi kajian yang dapat mengembangkan penelitian yang akan dilakukan.

Berikut adalah penelitian terdahulu yang dijadikan panduan atau acuan penulis dalam melakukan penelitian :

1. Penelitian yang dilakukan oleh Gea Lucita Hariyanti (2014).

" Sistem E-Learning Pada SMK Negeri 2 Baleendah Berbasis Web ". Tujuan penulisan adalah untuk menggambarkan sistem belajar yang sedang berjalan di SMK Negeri 2 Baleendah, membuat perancangan Sistem E-Learning Pada SMK Negeri 2 Baleendah Berbasis Web. Penulis menggunakan metodelogi pendekatan terstruktur meliputi diagram alir dokumen, diagram konteks dan diagram alir data. [3]

Persamaan dengan penelitian yang dilakukan oleh Gea Lucita Hariyanti yaitu membuat *e-learning* berbasis web pada sekolah menengah kejuruan dengan menggunakan metode pengembangan prototype. Perbedaannya yaitu penelirian tersebut menggunakan metode pendekatan sistem terstruktur.

2. Penelitian yang dilakukan oleh Sandy Kosasi (2015).

"Perancangan E-learning untuk Meningkatkan Motivasi Belajar Guru dan Siswa".Tujuan penelitian untuk menghasilkan sebuahrancangan aplikasi sistem E-learning berbasis web untuk membantu meningkatkan motivasi belajar guru dan siswa pada SMK Immanuel Pontianak. Meminimalisasi kesulitan komunikasi antara guru dan siswa dalam proses pembelajaran, memahami dan mengulang setiap materi pelajaran sekolah. Melalui aplikasi berbasis web memungkinkan guru dan siswa dapat berinteraksi dimanapun mereka berada tanpa batasanruangdanwaktu. Bentuk penelitian studi kasus menggunakan metode R&D (Research and Development). Instrumen penelitian melalui teknik wawancara dan menyebarkan angket kuesioner. Variabel penelitian meliputi interface, fungsionalitas dan fitur, penggunaan text dan gambar, perancangan dan validasi input, serta perancangan output. Untuk perancangan perangkat lunak menggunakan metode prototype requirement. Metode ini meliputi identifikasi kebutuhan pemakai, mengembangkan prototipe, menentukan prototipe diterima atau tidak, mengkodekan sistem operasional, menguji sistem operasional, menentukan apakah sistem operasional dapat diterima atau tidak, dan terakhir menggunakan sistem operasional.[4]

Persamaan dengan penelitian yang dilakukan oleh Sandy Kosasi yaitu membuat *e-learning* berbasis web yang memiliki fasilitas berbagi materi dan forum diskusi antara guru dengan siswa. Metodelogi yang digunakan adalah *prototype.*

2.2 Pengertian Sistem Informasi

Sistem dapat didefinisikan sebagai kumpulan dari komponen yang saling berhubungan satu dengan yang lainya membentuk satu kesatuan untuk mencapai tujuan tertentu. [5]

Informasi adalah data yang telah diklasifikasikan atau diolah atau diinterpretasikan untuk digunakan dalam proses pengambilan keputusan. [6]

Menurut Ais Zakiyudin, sistem informasi adalah suatu sistem yang ada di dalam suatu organisasi yang mempertemukan kebutuhan pengolahan transaksi harian, mendukung operasi yang bersifat manajerial dan kegiatan strategi dari suatu organisasi dan pihak luar tertentu dengan laporan yang diperlukan. [7]

Turban Mclean, dan Wetherbe (dalam Darmawan, 2013) menyatakan bahwa sistem informasi adalah sebuah sistem informasi yang mempunyai fungsi mengumpulkan, memproses, menyimpan, menganalisis, dan menyebarkan informasi untuk tujuan yang spesifik. [8]

Sedangkan Menurut penulis, sistem informasi adalah kumpulan elemen yang saling berhubungan satu sama lain yang membentuk satu kesatuan untuk mengintegrasikan data, memproses dan menyimpan, yang dimaksudkan untuk mencapai suatu tujuan tertentu.

2.1.1 Komponen Sistem Informasi

Menurut Yakub, sistem informasi merupakan sebuah susunan yang terdiri dari beberapa komponen atau elemen. [9] Komponen-komponen dari sistem informasi ini dapat digambarkan sebagai berikut ini :

1. Komponen Masukan (*Input Block)*

Input merupakan data yang masuk ke dalam sistem informasi. Input termasuk dalam metode dan media untuk menangkap data yang akan dimasukkan, data dapat berupa dokumen dokumen dasar.

2. Komponen Model (*Model Block*)

Kombinasi prosedur, logika, dan model matemetik yang akan memanipulasi data input dan data yang tersimpan di basis data dengan cara yang sudah ditentukan untuk menghasilkan keluaran yang diingiinkan.

3. Komponen Keluaran (*Output Block*)

Hasil dari blok keluaran yang merupakan informasi yang berkualitas dan dokumentasi yang berguna untuk semua tingkatan manajemen serta semua pemakai sistem.

4. Komponen Teknologi (*Technology Block*)

Teknologi merupakan kotak alat (*tool box*) dalam sistem informasi. Teknologi digunakan untuk menerima input, menjalankan model, menyimpan, dan

mengaksesdata, menghasilkan dan mengirimkan keluaran dan membantu pengendalian dari sistem secara menyeluruh.

5. Blok Basis Data (*Database Block*)

Merupakan kumpulan dari data yang saling berhubungan satu sama lainnya, tersimpan di perangkat keras komputer dan digunakan perangkat lunak untuk memanipulasinya.

2.1.2 Analisis Kebutuhan Sistem

Pembangunan sistem informasi memerlukan penyelidikan dan analisis mengenai alasan timbulnya ide atau gagasan untuk membangun dan mengembangkan sistem informasi. Analisis dilakukan untuk melihat berbagai komponen yang dipakai sistem yang sedang berjalan meliputi *hardware,software,* jaringan dan sumber daya manusia. Analisis juga mendokumentasikan aktivitas sistem informasi meliputi *input,* pemrosesan, *output,* penyimpanan dan pengendalian.

Selanjutnya melakukan studi kelayakan *(feasibility study)* untuk merumuskan informasi yang dibutuhkan pemakai akhir, kebutuhan sumber daya, biaya, manfaat dan kelayakan proyek yang diusulkan.

Analisis kebutuhan sistem sebagai bagian dari studi awal bertujuan mengidentifikasi masalah dan kebutuhan spesifik sistem. Kebutuhan spesifik sistem adalah spesifikasi mengenai hal-hal yang akan dilakukan sistem ketika diimplementasikan. [10]

Analisis kebutuhan sistem harus mendefinisikan kebutuhan sistem yang spesifik antara lain :

1) Masukan yang diperlukan sistem *(input)*
2) Keluaran yang dihasilkan *(output)*
3) Operasi-operasi yang dilakukan (proses)
4) Sumber data yang ditangani
5) Pengendalian (kontrol)

Tahap analisis kebutuhan sistem memerlukan evaluasi untuk mengetahui kemampuan sistem dengan mendefinisikan apa yang seharusnya dapat dilakukan oleh sistem tersebut kemudian menentukan kriteria yang harus dipenuhi sistem. Beberapa kriteria yang harus dipenuhi adalah pencapaian tujuan, kecepatan, biaya, kualitas informasi yang dihasilkan, efisiensi dan produktivitas, ketelitian dan validitas dan kehandalan atau reliabilitas.

2.1.3 Desain Sistem

Menurut Burch dan Grundnitski, desain sistem dapat didefinisikan sebagai penggmbaran, perencanaan, dan pembuatan sketsa atau pengaturan dari beberapa elemen yang terpisah kedalam suatu kesatuan yang utuh dan berfungsi.

Desain sistem menentukan bagaimana suatu sistem akan menyelesaikan tahap ini menyangkut konfigurasi dari komponen-komponen perangkat lunak dan perangkat keras dari suatu sistem sehingga setelah instalasi dari sistem akan benar-benar memuaskan rancang bangun yang telah ditetapkan pada akhir tahap analisis sistem. [11]

2.1.4 Implementasi Sistem

Setelah prototipe diterima maka pada tahap ini merupakan implementasi sistem yang siap dioperasikan dan selanjutnya terjadi proses pembelajaran terhadap sistem baru dan membandingkannya dengan sistem lama, evaluasi secara teknis dan operasional serta interaksi pengguna, sistem dan teknologi informasi.

2.1.5 Pengujian : *Black Box Testing*

Menurut Black, tester menggunakan behavioral test (disebut juga Black Box Tests), sering digunakan untuk menemukan bug dalam high level operations, pada tingkatan fitur, profil operasional dan scenario customer. Tester dapat membuat pengujian fungsional black box berdasrkan pada apa yang harus sistem lakukan. [12].

Menurut Nidhra dan Dondeti, black box testing juga disebut functional testing, sebuah teknik pengujian fungsional yang merancang test case berdasarkan informasi dari spesifikasi. [13]

Pengujian *black box* adalah pengujian aspek fundamental sistem tanpa memperhatikan struktur logika internal perangkat lunak . Pengujian *black-box* merupakan pendekatan komplementer yang kemungkinan besar mampu mengungkap kelas kesalahan. Pengujian *black-box* berusaha menemukan kesalahan dalam kategori sebagai berikut :

a. Fungsi-fungsi yang tidak benar atau hilang.
b. Kesalahan *interface*.
c. Kesalahan dalam struktur data atau akses *database* eksternal.

d. Kesalahan kinerja.

e. Inisialisasi dan kesalahan terminasi

Menurut Pressman, pengujian *black box* berfokus kepada persyaratan fungsional perangkat lunak. Pengujian *black box*memungkinkan perangkat lunak mendapatkan serangkaian kondisi input yang sepenuhnya menggunakan semua persyaratan fungsional untuk suatu program.[14] Pengujian *black box*bukan merupakan alternatif dari teknik *whitebox*, tetapi merupakan pendekatan komplementer yang kemungkinan besar mampu mengungkap kesalahan-kesalahan pada metode *white box*.

2.3 *E-learning*

E-learning dalam arti luas bisa mencakup pembelajaran yang dilakukan di media elektronik (internet) baik secara formal maupun informal. *E-learning* secara formal, misalnya adalah pembelajaran dengan kurikulum, silabus, mata pelajaran dan tes yang telah diatur dan disusun berdasarkan jadwal yang telah disepakati pihak-pihak terkait (pengelola *e-learning* dan pembelajaran sendiri). Pembelajaran seperti ini biasanya tingkat interaksinya tinggi dan diwajibkan oleh perusahaan pada karyawannya atau pembelajaran jarak jauh yang dikelola oleh universitas, sekolah dan perusahaan-perusahaan yang memang bergerak dibidang penyedian jasa *e-learning* untuk umum. *E-learning* bisa juga dilakukan secara informal dengan interaksi yang lebih sederhana, misalnya melalui sarana mailing *list*, *e-newsletter* atau *website* pribadi, organisasi dan perusahaan yang ingin

mensosialisasikan jasa, program, pengetahuan atau keterampilan tertentu pada masyarakat luas.

E-learning menurut Michael, merupakan pembelajaran yang disusun dengan tujuan menggunakan sistem elektronik atau komputer sehingga mampu mendukung proses pembelajaran.[15]

Sedangkan menurut adalah Ardiansyah, *e-learning* adalah sistem pembelajaran yang digunakan sebagai sarana untuk proses belajar mengajar yang dilaksanakan tanpa harus bertatap muka secara langsung antara guru dengan siswa.[16]

Manfaat *e-learning* dapat dilihat dari 2 sudut pandang:

a. Manfaat bagi siswa/i

Dengan kegiatan e-learning dimungkinkan berkembangnya fleksibilitas belajar yang tinggi. Artinya, kita dapat mengaskses bahan-bahan belajar setiap saat dan berulang-ulang. Selain itu, kita juga dapat berkomunikasi dengan guru setiap saat, misalnya melalui chatting dan email. Mengingat sumber belajar yang sudah dikemas secar elektronik dan tersedia untuk diakses melalui internet, maka kita dapat melakukan interaksi dengan sumber belajar ini kapan saja dan dimana saja, dan juga tugas dapat diserahkan kepada guru begitu selesai dikerjakan.

b. Manfaat bagi pengajar

Dengan adanya kegiatan *e-learning* manfaat yang diperoleh guru adalah guru akan lebih mudah melakukan pembaharuan materi maupun model pengajaran

sesuai dengan tuntutan perkembangan keilmuan yang terjadi, juga dapat efisien mengontrol kegiatan siswa/inya.

Maka dapat disimpulkan *e-learning* merupakan cara baru dalam proses belajar mengajar yang memanfaatkan media elektronik khususnya internet sebagai sistem pembelajarannya, sehingga proses belajar mengajar dapat dilakukan dimana saja dan kapan saja.

2.4 Pemodelan : *Object Oriented Programming*

Object-Oriented Programming (OOP) adalah sebuah pendekatan untuk pengembangan suatu *software* dimana dalam struktur software tersebut didasarkan kepada interaksi objek dalam penyelesaian suatu proses/tugas. Interaksi tersebutmengambil form dari pesan-pesan dan mengirimkannya kembali antar object tersebut. Objek akan merespon pesan tersebut menjadi sebuah tindakan /*action* atau metode.

Object Oriented Programming (OOP) adalah suatu metode pemrograman yang berorientasi kepada objek. Tujuan dari OOP diciptakan adalah untuk mempermudah pengembangan program dengan cara mengikuti model yang telah ada di kehidupan sehari-hari. Jadi, setiap bagian dari suatu permasalahan adalah objek, objek itu sendiri merupakan gabungan dari beberapa objek yang lebih kecil lagi.Konsep dari OOP adalah lebih dari sekedar sebuahkonsep pemrograman, OOP adalah cara berpikir tentang aplikasiyang mempelajari untuk berpikir bahwa aplikasi bukan sekedar prosedur melainkan sebagai objek dan *real entity*.[17]

2.5 *Unified Modeling Language* (UML)

Menurut Nugroho, *Unified modeling language*(UML) 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.[18]

MenurutWhitten dan Bentley, *Unified Modeling Language* adalah sekumpulan aturan yang digunakan untuk menspesifikasikan atau mendeskripsikan sebuah sistem perangkat lunak berdasarkan obyek. [19]

UML Diagram bisa diibaratkan seperti cetakan biru untuk membangun sebuah rumah. Satu set cetakan biru biasanya membantu pembangunnya dengan gambaran yang jelas untuk saluran air, listrik, pemanas, dan sejenisnya, setiap UML Diagram membantu tim developer program dengan gambaran yang jelas untuk sistem tersebut.

1) *Use Case Diagram*

Use case adalah kegiatan atau urutan interaksi yang saling berkaitan antara sistem dan aktor. *Use case* bekerja dengan cara mendeskripsikan tipe interaksi antara user sebuah sistem dengan sistemnya sendiri melalui sebuah cerita bagaimana sebuah sistem dipakai. *Use case* juga digunakan untuk membentuk perilaku (*behaviour*) sistem yang akan dibuat. Sebuah *use case* menggambarkan sebuah interkasi antara pengguna (aktor) dengan sistem yang ada.

Tujuan Use Case :

a. Memetakkan kebutuhan sistem

b. Merepresentasikan interaksi pengguna terhadap sistem

c. Untuk mengetahui kebutuhan diluar sistem

Komponen *use case diagram*:

a. Aktor

Aktor merupakan hal terpenting dari *use case diagram*, akan tetapi aktor bukanlah bagian dari *use case diagram*, namun untuk dapat terciptanya suatu *use case diagram* diperlukan beberapa aktor. Itulah pengertian Actor pada komponen *use case diagram* secara singkat .

b. *Use Case*

Use case merupakan sebuah gambaran fungsionalitas dari suatu sistem, sehingga pengguna sistem tersebut paham secara jelas mengenai kegunaan sistem atau aplikasi yang akan dibangun. Itulah pengertian *use case* pada komponen *use case diagram* secara singkat.

c. Relasi

Use case diagram memiliki relasi sebagai berikut:

1) *Association* : Menghubungkan link antar element.
2) *Generalization* : Sebuah elemen dapat merupakan spesialis dari elemen lainnya bisa disebut dengan pewarisan sifat (*inheritance*).

3) *Dependency* : Sebuah elemen bergantung dalam beberapa cara ke element lain.
4) *Aggregation* : Bentuk association dimana sebuah elemen berisi elemen lainnya.

Tipe Relasi/ Streotype yang mungkin terjadi pada *use case* diagram :

1) <<*include*>>, kelakuan yang harus terpenuhi agar sebuah *event* dapat terjadi.
2) <<*extends*>>, kelakuan yang hanya berjalan pada kondisi tertentu.
3) <<*communicates*>>, mungkin ditambahkan untuk asosiasi yang menunjukkan asosiasinya adalah *communicates association*.

d. *System Boundary*

Digambarkan dengan kotak disekitar *use case*, untuk menggambarkan jangkauan sistem anda (*scope of of your system*). Biasanya digunakan apabila memberikan beberapa alternative system yang dapat dijadikan pilihan. *System boundary boxes* dalam penggunaannya optional.

2) Skenario *Use Case*

Skenario *use case* digunakan untuk memudahkan dalam menganalisa skenario yang akan kita gunakan pada fase-fase selanjutnya dengan melakukan penilaian terhadap skenario tersebut.

3) *Activity Diagram*

Activity Diagram adalah diagram yang menggambarkan worlflow (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Yang perlu diperhatikan adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem.

Komponen-komponen yang ada pada activity diagram adalah sebagai berikut:

a. Nodes (initial dan final) adalah simbol untuk memulai (initial) dan mengakhiri (final) suatu activity diagram.
b. Activity (aktifitas) adalah proses komputasi yang bisa berupa kata kerja atau ekspresi dan bersifat atomiak atau todak dapat didekomposisi
c. Flow adalah awal dari proses yang paralel dan mampu menggambarkan aktivitas yang mungkin terjadi secara concurrent.
d. Join adalah akhir dari suatu proses paralel.
e. Decision adalah pilihan untuk mengambil keputusan.
f. Partition digunakan untuk menjelaskan siapa yang melakukan aktivitas dalam activity diagram. Untuk melakukan partisi dapat dilakukan dengan menggunakan *Swim Lane*.
g. Signal adalah tanda untuk memulai sebuah aktivitas.

4) *Sequence Diagram*

Sequence diagram adalah diagram yang 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 *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu. Banyaknya diagram sekuen yang harus digambar sebanyak usecase yang memiliki proses sendiri atau yang penting semua *use case* yang telah didefinisikan interaksi jalannya pesan sudah dicakup pada diagram sekuen sehingga semakin banyak usecase yang ada maka diagram sekuen yang dibuat semakin banyak.

Komponen *Sequence Diagram :*

a. Aktor :Menggambarkan seseorang atau sesuatu (seperti perangkat, sistem lain) yang berinteraksi dengan sistem.
b. *Boundary :*Mengambarkan interaksi antara satu atau lebih aktor dengan sistem, memodelkan bagian dari sistem yang bergantung pada pihak lain disekitarnya dan merupakan pembatas sistem dengan dunia luar.
c. *Control :*Menggambarkan,mengkoordinasikan perilaku sistem dan dinamika dari suatu sistem, menangani tugas utama dan mengontrol alur kerja suatu sistem.
d. *Entity :*Menggambarkan informasi yang harus disimpan oleh sistem (struktur data dari sebuah sistem).
e. *Object Message :*Menggambarkan pesan/hubungan antar obyek yang menunjukkan urutan kejadian yang terjadi.
f. *Message to Self :*Mengambarkan pesan/hubungan obyek itu sendiri, yang menunjukkan urutan kejadian yang terjadi.
g. *Return Message :*Menggambarkan pesan/hubungan antar obyek, yang menunjukan urutan kejadian yang terjadi.

h. *Lifeline :*Eksekusi obyek selama *sequence* (*message* dikirim atau diterima dan aktifasinya).

5) *Class Diagram*

Class Diagram adalah diagram yang menunjukan class-class yang ada dari sebuah sistem dan hubungannya secara logika. *Class diagram* menggambarkan struktur statis dari sebuah sistem. Karena itu class diagram merupakan tulang punggung atau kekuatan dasar dari hampir setiap metode berorientasi objek termasuk UML. [20]

Menurut Rosa A.S dan M. Shalahuddin,*class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. [1]

Komponen*Class Diagram :*

a. *Class* :

Blok - blok pembangun pada pemrograman berorientasi obyek. Sebuah *class* digambarkan sebagai sebuah kotak yang terbagi atas 3 bagian. Bagian atas adalah bagian nama dari *class*. Bagian tengah mendefinisikan *property*/atribut *class*. Bagian akhir mendefinisikan methodmethod dari sebuah *class*.

b. Association :

Sebuah asosiasi merupakan sebuah relationship paling umum antara 2 class dan dilambangkan oleh sebuah garis yang menghubungkan antara 2 class. Garis ini bisa melambangkan tipe-tipe relationship dan juga dapat menampilkan hokum -

hukum multiplisitas pada sebuah *relationship*.(Contoh:One-to-one, one-to-many, many-to-many).

c. Composition :

Jika sebuah *class* tidak bisa berdiri sendiri dan harus merupakan bagian dari *class* yang lain, maka class tersebut memiliki relasi *composition* terhadap *class* tempat dia bergantung tersebut. Sebuah *relationship composition* digambarkan sebagai garis dengan ujung berbentuk jajaran genjang berisi/solid.

d. Dependency :

Kadangkala sebuah *class* menggunakan class yang lain. Hal ini disebut *dependency*. Umumnya penggunaan *dependency* digunakan untuk menunjukkan operasi pada suatu *class* yang menggunakan *class* yang lain.

e. Aggregation :

Mengindikasikan keseluruhan bagian *relationship* dan biasanya disebut sebagai relasi.

6) *Object Diagram*

Diagram Objek menggambarkan struktur sistem dari segi penamaan objek dan jalannya objek dalam sistem. dalam diagram objek harus di pastikan bahwa semua kelas yang sudah didefinisikan pada diagram kelas harus dipakai objeknya, karena jika tidak pendefinisian kelas itu tidak dapat dipertanggung jawabkan.

Sebuah diagram objek merupakan gambaran objek-objek pada sebuah sistem pada satu titik waktu. karena lebih menonjolkan printah-printah dari pada kelas daigram objek sering disebut juga sebagai diagram perintah. elemen-elemen sebuah diagram objek adalah spesifikasi perintah.

Elemen-elemen diagram objek terdiri dari:

a. Objek merupakan Objek dari *class* yang berjalan saat sistem dijalankan
b. *Link*merupakan relasi antar objek

Terdapat dua *stereotype* terhadap *dependency* di antara objek, yaitu:

a. *Become,* menspesifikasikan target adalah objek yang sama dengan sumber, tapi pada titik waktu berikutnya dan dapat mempunyai nilai, *state* dan peran yang berbeda.
b. *Call,* menspesifikasikan operasi sumber memanggil operasi target.
c. *Copy*, menspesifikasikan objek target adalah nyata, tapi independen, mengkopi sumber.

7) *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.

Komponen *Deployment Diagram :*

a. *Component :*Pada *deployment diagram*, komponen-komponen yang ada diletakkan didalam node untuk memastikan keberadaan posisi mereka.

b. *Node :Node* menggambarkan bagian-bagian*hardware* dalam sebuah sistem. Notasi untuk node digambarkan sebagai sebuah kubus 3 dimensi.

c. *Association :*Sebuah *association* digambarkan sebagai sebuah garis yang menghhubungkan dua node yang mengindikasikan jalur komunikasi antara komponen - komponen *hardware*.

d. *Dependency :*Merupakan relasi yang menunjukan bahwa perubahan pada salah satu elemen memberi pengaruh pada elemen lain.

e. *Generalization :*Menunjukan hubungan antara elemen yang lebih umum ke elemen yang lebih spesifik. Dengan *generalization, class* yang lebih spesifik (*subclass*) akan menurunkan atribut dan operasi dari *class* yang lebih umum (*superclass*).

f. *Note :*Menunjukan catatan untuk komentar dari suatu pesan antar elemen, sehingga bisa langsung terlampir dalam model.

g. *Class :Interface* merupakan kumpulan operasi tanpa implementasi dari suatu *class*. Implementasi operasi dalam *interface* dijabarkan oleh operasi didalam *class*.

8) *Component Diagram*

C*omponent diagram* dibuat untuk menunjukkan organisasi dan ketergantungan diantara kumpulan komponen dalam sebuah sistem. diagram komponen fokus pada komponen sistem yang dibutuhkan dan ada didalam sistem. diagram komponen juga dapat digunakan untuk memodelkan hal-hal berikut :

a. *Source code* program perangkat lunak.
b. Komponen *executable* yang di lepas ke *user.*
c. Basis data secara fisik.
d. Sistem yang harus beradaptasi dengan sistem lain.
e. *Framework* sistem, *framework* pada perangkat lunak merupakan kerangka kerja yang dibuat untuk memudahkan pengembangan dan pemeliharaan aplikasi, contohnya seperti struts dari apache yang menggunakan prinsip desain *Model-View-Controller* (MVC) dimana *soucre code* program dikelompokkean berdasarkan fungsinya. dimana *controller* berisi *source code* yang menangani request dan validasi, model berisi *source code* yang menangani manipulasi data dan *bussiness logic*, dan *view* berisi *source code* yang menangani tampilan.

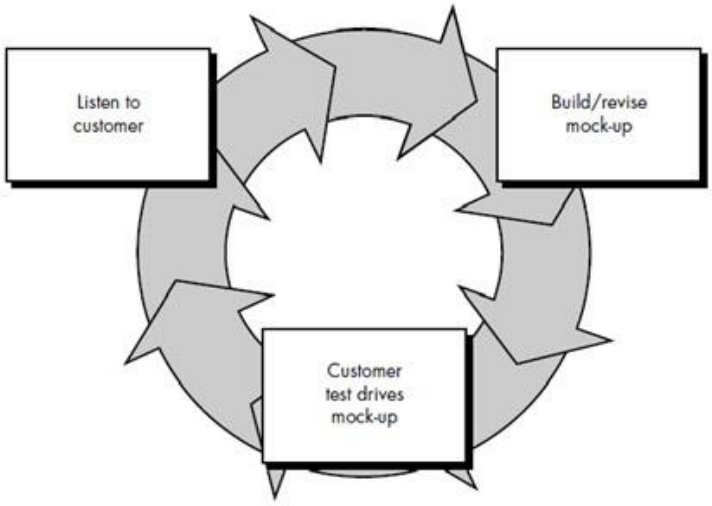
Komponen dasar yang biasanya ada dalam suatu sistem adalah sebagai berikut :

a. Komponen *user interface* yang menangani tampilan.
b. Komponen *bussiness process* yang menangani fungsi-fungsi proses bisnis.
c. Komponen data yang menangani manipulasi data.
d. Komponen *security* yang menangani keamanan sistem.

2.6 Metode Pengembangan Perangkat Lunak : *Prototype*

Menurut Roger S. Pressman, dalam melakukan perancangan sistem yang akan dikembangkan dapat mengunakan metode *prototype*. Metode ini cocok digunakan untuk mengembangkan sebuah perangkat yang akan dikembangkan kembali. Metode ini dimulai dengan pengumpulan kebutuhan pengguna, dalam hal ini pengguna dari perangkat yang dikembangkan adalah peserta didik. Kemudian membuat sebuah rancangan kilat yang selanjutnya akan dievaluasi kembali sebelum diproduksi secarabenar. [14]

Prototype bukanlah merupakan sesuatu yang lengkap, tetapi sesuatu yang harus dievaluasi dan dimodifikasi kembali. Segala perubahan dapat terjadi pada saat *prototype* dibuat untuk memenuhi kebutuhan pengguna dan pada saat yang sama memungkinkan pengembang untuk lebih memahami kebutuhan pengguna secara lebih baik.



Gambar 2. 1Paradigma Pembuatan Prototype

(Sumber : Roger, S. Pressman, Ph.D. Rekayasa Perangkat Lunak (Pendekatan Praktisi) Edisi 7 : Buku 1, 2012)

Gambar model *prototype* menurut Roger S. Pressman [14]:

1. Mendengarkan Pelanggan

Pada tahap ini dilakukan pengumpulan kebutuhan dari sistem dengan cara mendengar kebutuhan dari pelanggan. Untuk membuat suatu sistem yang sesuai kebutuhan, maka harus diketahui terlebih dahulu bagaimana sistem yang sedang berjalan untuk kemudian mengetahui masalah yang terjadi.

2. Merancang dan Membuat *Prototype*

Pada tahap ini, dilakukan perancangan dan pembuatan *prototype system*. *Prototype* yang dibuat disesuaikan dengan kebutuhan sistem yang telah didefinisikan sebelumnya dari kebutuhan pelanggan atau pengguna.

3. Uji Coba

Pada tahap ini, *prototype* dari sistem di uji coba oleh pelanggan atau pengguna. Kemudian dilakukan evaluasi kekurangan-kekurangan dari kebutuhan pelanggan. Pengembangan kemudian kembali mendengarkan keluhan dari pelanggan untuk memperbaiki *prototype* yang ada.

2.7 Perangkat Lunak Pendukung

Penelitian ini menggunakan beberapa perangkat lunak untuk membangun *e-learning* di SMK Negeri 1 Bandung. Berikut adalah urutan perangkat lunak yang digunakan dalam membangun*e-learning*tersebut :

2.7.1 Laravel

Laravel adalah sebuah *framework* PHP yang dirilis dibawah lisensi MIT, dibangun dengan konsep MVC (model view controller). Laravel adalah pengembangan *website* berbasis MVP yang ditulis dalam PHP yang dirancang untuk meningkatkan kualitas perangkat lunak dengan mengurangi biaya pengembangan awal dan biaya pemeliharaan, dan untuk meningkatkan pengalaman bekerja dengan aplikasi dengan menyediakan *sintaks* yang ekspresif, jelas dan menghemat waktu.

MVC adalah sebuah pendekatan perangkat lunak yang memisahkan aplikasi logika dari presentasi. MVC memisahkan aplikasi berdasarkan komponen-komponen aplikasi, seperti : manipulasi data, *controller*, dan *user interface*.

1) Model, Model mewakili struktur data. Biasanya model berisi fungsi-fungsi yang membantu seseorang dalam pengelolaan basis data seperti memasukkan data ke basis data, pembaruan data dan lain-lain.
2) *View, View* adalah bagian yang mengatur tampilan ke pengguna. Bisa dikatakan berupa halaman web.
3) *Controller, Controller* merupakan bagian yang menjembatani model dan *view*.

Beberapa fitur yang terdapat di Laravel:

1) *Bundles*, yaitu sebuah fitur dengan sistem pengemasan modular dan tersedia beragam di aplikasi.
2) *Eloquent* ORM, merupakan penerapan PHP lanjutan menyediakan metode *internal* dari pola "*active record*" yang menagatasi masalah pada hubungan objek *database*.
3) *Application Logic*, merupakan bagian dari aplikasi, menggunakan *controller* atau bagian *Route*.
4) *Reverse Routing*, mendefinisikan relasi atau hubungan antara *Link* dan *Route*.
5) *Restful controllers*, memisahkan logika dalam melayani HTTP GET and POST.
6) *Class Auto Loading*, menyediakan *loading* otomatis untuk *class* PHP.
7) *View Composer*, adalah kode unit logikal yang dapat dieksekusi ketika *view* sedang *loading*.

8) *IoC Container*, memungkin obyek baru dihasilkan dengan pembalikan *controller*.
9) *Migration*, menyediakan sistem kontrol untuk skema *database*.
10) *Unit Testing*, banyak tes untuk mendeteksi dan mencegah regresi.
11) *Automatic Pagination*, menyederhanakan tugas dari penerapan halaman.

2.7.2 Perl Hypertext Preprocessor (PHP)

Menurut Arief, *PHP (Perl Hypertext Preprocessor)* adalah bahasa server-side-scripting yang menyatu dengan HTML untuk membuat halaman web yang dinamis. [21] Dengan menggunakan program PHP, sebuah *website* akan lebih interaktif dan dinamis.

Adapun kelebihan-kelebihan dari PHP yaitu:

1) PHP merupakan sebuah bahasa *script* yang tidak melakukan sebuah kompilasi dalam penggunaannya. Tidak seperti halnya bahasa pemrograman aplikasi yang lainnya.
2) PHP dapat berjalan pada *web server* yang dirilis oleh Microsoft, seperti IIS atau PWS juga pada apache yang bersifat *open source*.
3) Karena sifatnya yang *open source*, maka perubahan dan perkembangan interpreter pada PHP lebih cepat dan mudah, karena banyak milis-milis dan *developer* yang siap membantu pengembanganya.
4) Jika dilihat dari segi pemahaman, PHP memiliki referensi yang begitu banyak sehingga sangat mudah untuk dipahami.

5) PHP dapat berjalan pada 3 operating sistem, yaitu: Linux, unux, dan windows, dan juga dapat dijalankan secara *runtime* pada suatu *console*.

2.7.3 Bootstrap

Bootstrap adalah sebuah *framework* CSS yang menyediakan kumpulan komponen-komponen antarmuka dasar pada web yang telah dirancang sedemikian rupa untuk digunakan bersama-sama. Selain komponen antarmuka, Bootstrap juga menyediakan sarana untukmembangun*layout* halaman dengan mudah dan rapi, serta modifikasi pada tampilan dasar HTML untuk membuat seluruh halaman web yang dikembangkan senada dengan komponen-komponen lainnya.Bootstrap dibuat untuk memberikan sekumpulan perangkat yang dapat digunakan untuk membangun *website* sederhana dengan mudah.

2.7.4 Cascading Style Sheet (CSS)

CSS merupakan salah satu kode pemrograman yang bertujuan untuk menghias dan mengatur gaya tampilan/layout halaman web supaya lebih elegan dan menarik.CSS adalah sebuah teknologi internet yang direkomendasikan oleh World Wide Web Consortium atau W3C pada tahun 1996. Awalnya, CSS dikembangkan di SGML pada tahun 1970, dan terus dikembangkan hingga saat ini. CSS telah mendukung banyak bahasa markup seperti HTML, XHTML, XML, SVG (Scalable Vector Graphics) dan Mozilla XUL (XML User Interface Language).

Pada desember 1996, W3C memperkenalkan Level 1 spesifikasi CSS atau juga dikenal CSS1 yang mendukung format, warna font teks, dan lain-lain.

Kemudian, Mei 1998, W3C menerbitkan CSS2 yang di dalamnya diatur fungsi peletakan elemen. Dan sekarang, W3C telah memperbaiki dan meningkatkan Kemampuan CSS2 ke CSS3.

CSS digunakan oleh web programmer dan juga blogger untuk menentukan warna, tata letak font, dan semua aspek lain dari presentasi dokumen di situs mereka. Saat ini, hampir tidak ada situs web yang dibangun tanpa kode css.

2.7.5 Database Management System (DBMS)

DBMS adalah singkatan dari "*Database Management System*" yaitu sistem pengorganisasian dan sistem pengolahan *database* pada komputer. DBMS atau *database management system* ini merupakan perangkat lunak (*software*) yang dipakai untuk membangun basis data yang berbasis komputerisasi.

DBMS (*Database Management system*) ini juga dapat membantu dalam memelihara serta pengolahan data dalam jumlah yang besar, dengan menggunakan DBMS bertujuan agar tidak dapat menimbulkan kekacauan dan dapat dipakai oleh *user* sesuai dengan kebutuhan.

DBMS ialah perantara untuk user dengan basis data, untuk dapat berinteraksi dengan DBMS dapat memakai bahasa basis data yang sudah di tentukan oleh perusahaan DBMS. Bahasa basis data umumnya terdiri dari berbagai macam instruksi yang diformulasikan sehingga instruksi tersebut dapat di proses oleh DBMS.

Perintah atau instruksi tersebut umumnya ditentukan oleh *user*, adapun bahasa yang digunakan dibagi kedalam 2 (dua) macam diantaranya sebagaimana di bawah ini:

1) DDL (*Data Definition Language*)

Yang pertama adalah bahasa DDL atau kepanjangannya *Data Definition Languange*, yaitu dipakai untuk menggambarkan desain dari basis data secara menyeluruh. DDL (*Data Definition Language*) dapat dipakai untuk membuat tabel baru, memuat indeks, maupun mengubah tabel. Hasil dari kompilasi DDL akan disimpan di kamus data. Itulah definisi dari DDL.

2) DML (*Data Manipulation Language*)

DML atau kepanjangannya *Data Manipulation Language*, yaitu dipakai untuk memanipulasi daan pengambilan data pada suatu basis data, misalnya seperti penambahan data yang baru ke dalam suatu basis data, menghapus data pada suatu basis data dan mengubah data pada suatu basis data. Itulah definisi dar DML.

2.7.6 MySQL

MySQL merupakan DBMS yang bersifat *open source* dimana pengguna dapat melakukan modifikasi pada kodenya yang berorientasi pada performa dan merupakan DBMS yang mendukung beberapa tipe *table* (*storage engine*). Tipe *table* yang digunakan secara *default* adalah *MyISAM* yang memiliki kelebihan akses data yang cepat tetapi tidak mendukung fitur *foreign key*. Jika pengguna ingin ada dukungan foreign key pada *table*, pengguna dapat menggunakan tipe *table* InnoDB.

Kelebihannya:

1) *Free*/gratis.
2) Selalu stabil dan cukup tangguh.
3) Keamanan yang cukup baik.
4) Sangat mendukung transaksi, dan dukungan dari banyak komunitas.
5) Sangat fleksisbel dengan barbagai macam program.
6) Perkembangan yang cepat.

Kekurangannya:

1) Kurang mendukung koneksi bahasa pemerograman misalnya seperti Visual Basic (VB), Foxpro, Delphi sebab koneksi ini dapat menyebabkan field yang dibaca harus sesuai dengan koneksi bari bahasa pemerograman visual tersebut.
2) Data yang dapat ditangani belum besar dan belum mendukung widowing Function.

2.7.7 LAMP

LAMP adalah istilah yang merupakan singkatan dari Linux, Apache, MySQL, dan Perl/PHP/Phyton. Merupakan sebuah paket perangkat lunak bebas yang digunakan untuk menjalankan sebuah aplikasi secara lengkap.

Komponen-komponen dari LAMP :

1. **L**inux – sistem operasi
2. **A**pache HTTP Server – web server
3. **M**ariaDB atau **M**ySQL – sistem basis data

4. **P**HP atau **P**erl atau **P**ython – bahasa pemrograman yang dipakai

2.8 Arsitektur Aplikasi

Arsitektur aplikasi berisi penjelasan tentang jaringan komputer, jenis-jenis jaringan komputer, topologi jaringan komputer, dan manfaat jaringan Komputer.

2.8.1 Jaringan Komputer

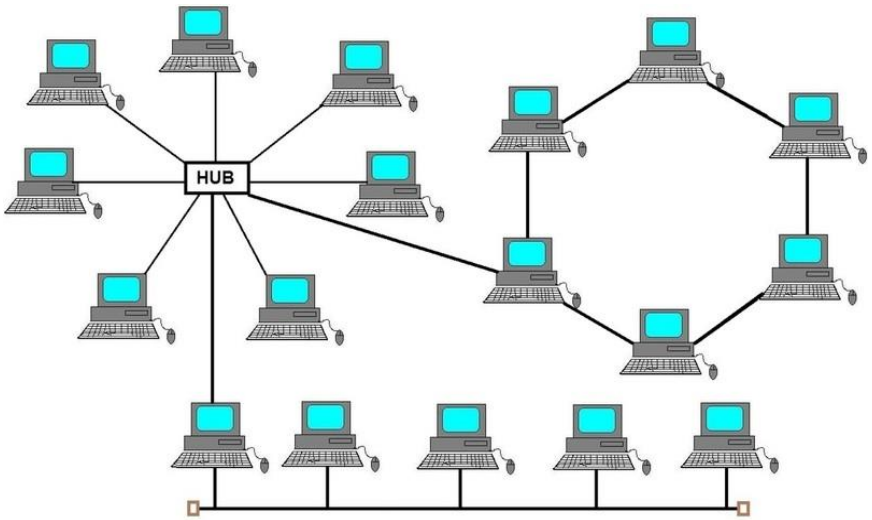
Jaringan komputer adalah suatu sistem yang didalamnya terdiri dari dua atau lebih perangkat komputer serta perangkat - perangkat lainnya yang dibuat atau dirancang untuk dapat berkerja sama dengan tujuan agar dapat berkomunikasi, mengakses informasi, meminta serta memberikan layanan atau *service* antara komputer satu dengan yang lainnya.

Menurut Jafar Noor Yudianto**,** jaringan komputer adalah sebuah sistem yang terdiri atas komputer-komputer yang didesain untuk dapat berbagi sumber daya (*printer*, CPU), berkomunikasi (surel, pesan instan), dan dapat mengakses informasi (peramban web). Tujuan dari jaringan komputer adalah agar dapat mencapai tujuannya, setiap bagian dari jaringan komputer dapat meminta dan memberikan layanan (*service*). Pihak yang meminta/menerima layanan disebut klien (*client*) dan yang memberikan/mengirim layanan disebut peladen (*server*). Desain ini disebut dengan sistem *client-server*, dan digunakan pada hampir seluruh aplikasi jaringan computer. [22]

Topologi jaringan adalah susunan fisik bagaimana node-node saling dihubungkan.Ada empat topologi yang biasa digunakan dalam membangun sebuah

jaringan, yaitu topologi *bus*, topologi *ring*, topologi s*tar* atau *hub*, dan topologi *hybrid*. Adapun topologi yang digunakan oleh penulis dalam perancangan *E-learning* pada SMK Negeri 1 Bandung adalah topologi *hybird*. Topologi *Hybrid* adalah salah satu bentuk topologi jaringan fisik yang sudah sangat umum digunakan disamping jenis komputer lainnya seperti jaringan *point-to-point*, topologi *bus*, topologi *ring*, dll. Klasifikasi topologi dibuat berdasarkan hubungan antara node berbeda dalam jaringan. Pilihan untuk menggunakan topologi tertentu tergantung pada berbagai faktor. Masing-masing topologi ini memiliki kelebihan dan kekurangan.

Topologi *Hybrid* merupakan salah satu topologi jaringan komputer yang populer karena kelebihan Topologi *Hybrid* memiliki nilai plus dibanding topologi lainnya dan sudah terbukti menjadi salah satu bentuk terbaik untuk kebutuhan jaringan komputer skala besar. Namunpun demikian tidak berarti topologi *hybrid* tidak memiliki kekurangan.



Gambar 2. 2 Topologi Jaringan Hybird