

## BAB II TINJAUAN PUSTAKA DAN LANDASAN TEORI

### 2.1 Tinjauan Pustaka

Penelitian tentang penerimaan magang yang dilakukan oleh A. Lestari and M. Novita dengan judul “ Sistem Informasi Magang Berbasis *Website* pada Dinas Kesehatan Provinsi Jawa Tengah”. Aplikasi ini merupakan suatu sistem yang digunakan untuk pengelolaan data peserta magang untuk melakukan pendaftaran, mendapatkan informasi bagian yang tersedia dan pengumuman hasil magang, bagian dan kuota pendaftar yang disediakan oleh instansi/perusahaan sehingga membuat proses pendaftaran magang lebih efektif dan efisien tanpa menanyakan langsung ke kantor/perusahaan. Metode pengembangan perangkat lunak yang dipakai adalah metode *waterfall*. Pengujian sistem ini menggunakan pengujian *black box* dan pengujian UAT (*User Acceptance Test*)[1].

Penelitian penerimaan magang juga dilakukan oleh H. Amelia and H. N. Irmanda dengan judul “ Sistem Informasi Magang pada Upt Teknologi Informasi dan Komunikasi Universitas Pembangunan Nasional Veteran Jakarta “. Tujuan dari penelitian ini adalah untuk memudahkan pendaftaran magang, pengelolaan data log harian dan kehadiran, serta pengelolaan nilai pemagang. Metode pengembangan perangkat lunak yang dipakai adalah metode *waterfall*. Penelitian ini menggunakan bahasa pemrograman PHP versi 7 dengan menggunakan *framework* codeigniter versi 3[2].

Penelitian lain tentang penerimaan magang juga dilakukan oleh M. I. M. Wihelvi dengan judul “ Sistem Informasi Magang Kerja Mahasiswa di Politeknik Tunas Pemuda Berbasis *Web*”. Tujuan dari penelitian ini adalah memberikan informasi seputar magang kerja, mulai darilowongan kerja, log harian bagi mahasiswa, informasi lokasi magang kerja, prosedur pengajuan magang kerja, dan jadwal ujian bagi mahasiswa yang sudah melaksanakan magang kerja. Perancangan sistem informasi magang kerja mahasiswa di Politeknik Tunas Pemuda berbasis *web* menggunakan metode *waterfall* [3].

Penelitian ini bermaksud membangun sistem yang berjudul “Sistem Informasi Penerimaan Magang Industri pada Sekretariat Daerah Kabupaten Cilacap Berbasis *Web*”. Perbedaan sistem ini dengan sistem-

sistem sebelumnya yaitu Metode pengembangan sistem sebelumnya menggunakan metode pengembangan sistem *waterfall* dan penulis bermaksud menggunakan metode pengembangan sistem *prototype*. Sistem ini juga dilengkapi dengan fitur *notifikasi Gmail* untuk memberikan pengumuman penerimaan magang yang lebih cepat dan efisien kepada para pendaftar magang di sekretariat daerah kabupaten Cilacap.

## **2.2 Landasan Teori**

### **2.2.1 Pengertian Informasi**

Pengertian informasi menurut Kusri dan Koniyo (2007) merupakan data yang sudah diolah menjadi sebuah bentuk yang berarti bagi pengguna, yang bermanfaat dalam pengambilan keputusan saat ini atau mendukung sumber informasi. Pengertian lain dari informasi menurut Susanto (2004) merupakan hasil dari pengolahan data yang memberikan arti dan manfaat. Dari pengertian diatas, dapat disimpulkan bahwa data harus diolah terlebih dahulu agar dapat menjadi informasi yang bermanfaat untuk pemakai informasi. Informasi yang berkualitas menurut Susanto (2008) harus memiliki ciri-ciri sebagai berikut: (1) Akurat, artinya informasi harus mencerminkan keadaan yang sebenarnya, (2) Tepat waktu, artinya informasi harus tersedia atau ada pada saat informasi tersebut diperlukan, (3) Relevan, artinya informasi yang diberikan harus sesuai dengan kebutuhan individu yang ada di berbagai tingkatan dan bagian dalam organisasi, dan (4) Lengkap, artinya informasi harus diberikan secara lengkap

### **2.2.2 Pengertian Sistem Informasi**

Pengertian sistem informasi merupakan satuan komponen yang saling berhubungan mengumpulkan, memproses, menyimpan, menganalisis, menyebarkan informasi untuk tujuan tertentu[4]. Kelebihan sistem informasi adalah sebagai penyedia data dan penyimpanan terbesar yang dapat memungkinkan penyelesaian proses kerja lebih cepat sehingga dapat membantu menyelesaikan permasalahan yang dihadapi. Sistem informasi juga dapat diartikan media yang diimplementasikan secara teknologi guna keperluan pencatatan, penyimpanan serta penyebaran aktualisasi kebahasaan dan guna mendukung pembuatan inferensi. Sistem informasi dapat didefinisikan sebagai kumpulan sistem-sistem yang saling terintegrasi satu dengan yang lainnya sehingga bermanfaat bagi pemakainya. Data akan tersimpan pada suatu database yang mengakibatkan data menjadi teratur dan memudahkan penggunaannya.

Berdasarkan referensi pengertian sistem informasi maka dapat disimpulkan bahwa sistem informasi adalah sebuah implementasi dari teknologi terkomputerisasi yang dapat mencatat, menyimpan dan menyebarkan sebuah informasi dimana fungsinya untuk memudahkan penggunaannya.

### 2.2.3 Rekayasa Web

Rekayasa *web* adalah proses yang digunakan untuk membuat aplikasi *web* berkualitas tinggi. Proses rekayasa *web* diawali dengan menentukan cara menyelesaikan masalah melalui aplikasi berbasis *web*. Aplikasi *web* adalah aplikasi berbasis web yang dibuat atau diimplementasikan oleh insinyur *web* dan pengembang non teknis[5]. Tahapan dalam rekayasa web yaitu *Communication, Planning, Modeling Construction, dan Deployment*.

1. *Customer communication*

Langkah paling efektif dalam memahami apa saja yang diinginkan oleh *user* adalah melalui komunikasi yang baik. Setelah mengetahui apa yang diinginkan oleh *user* maka dapat dibuat rumusan masalah terhadap sistem yang akan dibangun (*formulation*). Melakukan pengembangan sistem sesuai dengan kebutuhan memerlukan perjanjian dengan *user* dapat dilakukan setelah merumuskan masalah (*negatitation*). Kemudian observasi terhadap sistem yang akan dibangun (*elicitation*).

2. *Planning*

Perencanaan teknis dan wawancara merupakan hasil dari kebutuhan dan informasi yang diperoleh dari *user*. Mengidentifikasi perangkat lunak dan perangkat keras yang dibutuhkan adalah bagian dari perencanaan teknis. Perkiraan biaya pada sistem yang akan dibangun (*estimation*), menganalisis resiko oleh pengembang sistem (*risk analysis*), membuat jadwal sistem (*scheduling*) dan memantau sistem yang sedang dikembangkan (*monitoring*).

3. *Modeling*

Perancangan tampilan halaman dengan memperhatikan jenis teks dan warna sehingga dapat disesuaikan dengan alur kerja sistem. Selain merancang tampilan, pada tahap ini juga menyelesaikan masalah antara *user* dan fungsinya didalam sistem.

4. *Implementation(Coding)*  
Hasil perancangan aktivitas dituangkan dalam bentuk html untuk halaman *web* dan bentuk PHP untuk fungsi logika.
5. *Testing*  
Kesalahan-kesalahan yang mungkin terjadi pada sistem dapat diketahui pada tahap ini. Masalah yang dapat dideteksi mulai dari kesalahan tampilan hingga fungsi sistem.
6. *Deployment*  
Penilaian serta evaluasi terhadap sistem dan pengembang merupakan tahap akhir dari rekayasa *web*. Menyebarkan kuesioner kepada *user* untuk mendapat penilaian dari setiap kriteria yang ada didalamnya.

#### **2.2.4 Rekayasa Perangkat Lunak**

Perangkat lunak adalah seluruh perintah yang digunakan untuk memproses informasi. Perangkat lunak dapat berupa program atau prosedur. Program adalah kumpulan perintah yang dimengerti oleh komputer sedangkan prosedur adalah perintah yang dibutuhkan oleh pengguna dalam memproses informasi[6].

Rekayasa perangkat lunak sendiri adalah suatu disiplin ilmu yang membahas semua aspek produksi perangkat lunak, mulai dari tahap awal yaitu analisa kebutuhan pengguna, desain, pengkodean, pengujian, sampai pemeliharaan setelah digunakan.

##### **2.2.4.1 UML (*United Modelling Language*)**

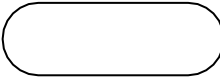


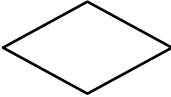

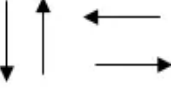

UML adalah salah satu *tool*/model untuk merancang pengembangan *software* yang berbasis *object-oriented*. UML memberikan standar penulisan sebuah sistem *blueprint*, yang meliputi konsep proses bisnis, penulisan kelas-kelas dalam bahasa program yang spesifik, skema *data base*, dan komponen yang diperlukan dalam sistem *software*[7].

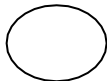
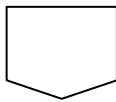
##### **2.2.4.1.1 Flowchart**

*Flowchart* merupakan penyajian yang sistematis tentang proses dan logika dari kegiatan penanganan informasi atau penggambaran secara grafik dari langkah-langkah dan urutan-urutan prosedur dari suatu program[10] . Bagan alir (*flowchart*) adalah bagan (*chart*) yang menunjukkan alir (*flow*) di dalam program atau prosedur sistem secara

logika. Bagan alir digunakan terutama untuk alat bantu komunikasi dan dokumentasi. Seperti pada Tabel 2.1.

**Tabel 2. 1** *Flowchart*




No	Simbol	Nama	Keterangan
1.		<i>Terminal</i>	Awal atau akhir sebuah program
2.		<i>Output/Input</i>	Proses input atau output
3.		Proses	Proses operasional di dalam komputer
4.		<i>Decision</i>	Suatu kondisi tertentu mengarah pada dua pilihan kemungkinan ya/tidak.
5.		<i>Document</i>	Simbol yang menunjukkan untuk mencetak laporan ke printer.
6.		<i>Flow Direction</i>	Simbol yang berfungsi untuk menyatakan jalannya alur sebuah proses
7.		<i>Manual</i>	Simbol yang menunjukkan proses diluar komputer atau kegiatan tanpa


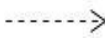
			penggunaan komputer
8.		<i>Magnetic-tape unit</i>	Simbol untuk menyatakan input berasal dari pita magnetic atau output disimpan ke pita magnetic.
9.		<i>Offline Connector</i>	Penghubung antar proses pada halaman yang lain

#### 2.2.4.1.2 Use case

*Use case* merupakan pemodelan untuk kelakuan sistem informasi yang akan dibuat[8]. *Use case* bekerja dengan mendeskripsikan tipikal interaksi antara user sebuah sistem dengan sistem nya sendiri melalui sebuah cerita bagaimana sistem itu dipakai. Seperti pada Tabel 2.2.

**Tabel 2. 2** *Use case* Diagram



No	Gambar	Nama	Keterangan
1.		<i>Actor</i>	Menspesifikasikan himpunan tokoh yang pengguna mainkan saat berhubungan menggunakan <i>use case</i> .
2.		<i>Include</i>	Menunjukkan bahwa suatu <i>use case</i> seluruhnya merupakan fungsionalitas dari <i>use case</i> lainnya.
3.		<i>Association</i>	Berguna untuk menghubungkan objek yang satu dengan objek yang lain

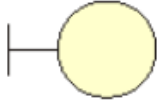



4.		<i>Use Case</i>	Deskripsi berdasarkan urutan aksi-aksi yang ditampilkan sistem yang membuat suatu output terukur bagi sebuah aktor
5.		<i>Extend</i>	Penghubung komunikasi antar aktor dan <i>use case</i>

#### 2.2.4.1.3 Sequence diagram

*Sequence* diagram digunakan untuk menggambarkan perilaku pada sebuah skenario. Diagram ini menunjukkan sejumlah objek dan message yang diletakan antara objek-objek di dalam *use case*. Komponen utama *sequence* diagram terdiri atas objek yang dituliskan dengan kotak segiempat bernama. *Message* diwakili oleh garis dengan tanda panah dan waktu yang ditunjukkan dengan *progress vertical*. *Sequence* diagram menambahkan dimensi waktu pada interaksi diantara objek [9]. Seperti pada Tabel 2.3.

**Tabel 2. 3** *Sequence* Diagram

No.	Nama	Simbol	Keterangan
1.	<i>Actor</i>		Menggambarkan orang yang berinteraksi dengan sistem
2.	<i>Entity Class</i>		Menggambarkan hubungan yang akan dilakukan





No.	Nama	Simbol	Keterangan
3.	<i>Boundary Class</i>		Menggambarkan sebuah gambaran dari <i>form</i>
4.	<i>Control Class</i>		Menggambarkan hubungan antara <i>boundary</i> dengan tabel
5.	<i>Lifeline</i>		Menggambarkan tempat mulai dan berakhirnya <i>message</i>
6.	Line Message		Menggambarkan pengiriman <i>message</i> antar <i>class</i>

#### 2.2.4.1.4 Entity Class Diagram

*Entity Relationship Diagram* (ERD) merupakan diagram yang berbentuk notasi grafis, yang berada dalam pembuatan database yang menghubungkan antara data satu dengan yang lain. Fungsi dari ERD adalah sebagai alat bantu dalam pembuatan database dan memberikan gambaran bagaimana kerja database yang akan dibuat. ERD memiliki 3 elemen dasar yaitu entitas, atribut, dan relasi. Berikut merupakan simbol-simbol ERD yang dapat dilihat pada tabel 2.4:



Tabel 2. 4 Tabel ERD

No.	Nama	Simbol	Keterangan Fungsi
1.	Entitas		Persegi panjang menyatakan himpunan entitas adalah orang, kejadian, atau berada.
2.	Atribut		Atribut merupakan informasi yang diambil tentang sebuah entitas.
3.	Relasi		Belah ketupat menyatakan himpunan relasi merupakan hubungan antar entitas.
4.	Associative Entity		Entitas yang digunakan pada relasi many-to-many.

### 2.2.5 System Development Prototype

Tahap pengembangan sistem yang akan dilakukan oleh penulis untuk membangun sistem ini yaitu menggunakan metode *Prototype*. Model *prototype* memberikan kesempatan untuk pengembang program dan objek penelitian untuk saling berinteraksi selama proses perancangan sistem. Metode pengembangan sistem *prototype* memiliki banyak kelebihan diantaranya yaitu; dapat menjalin komunikasi yang baik antar user dan pengembang, user akan memberikan masukan terhadap sistem sesuai dengan kemauannya, menghemat waktu dalam mengembangkan sebuah sistem, penerapan dari sistem menjadi lebih mudah untuk dilakukan [11].

### 2.2.6 Black Box Testing

Metode *Black box Testing* adalah metode pengujian yang berfokus pada spesifikasi fungsional dari perangkat lunak, tester dapat mendefinisikan kumpulan kondisi *input* dan melakukan pengetesan pada spesifikasi fungsional program[12]. Keuntungan menggunakan *black box testing* adalah [13] :

1. Penguji Penguji tidak perlu memiliki pengetahuan tentang Bahasa pemrograman tertentu.
2. Pengujian dilakukan dari sudut pandang pengguna, ini membantu untuk mengungkapkan ambiguitas atau inkonsistensi dalam spesifikasi persyaratan.
3. Programmer dan tester keduanya saling tergantung.

Kekurangan menggunakan *black box testing* adalah[14] :

1. Pengujian tidak bisa dilakukan sepenuhnya dikarenakan pengetahuan penguji terbatas tentang perangkat lunak yang diuji.
2. Uji kasus sulit dirancang tanpa spesifikasi yang jelas.

### 2.2.7 Pemrograman Berbasis Objek

Metodologi berorientasi objek adalah strategi pembangunan perangkat lunak sebagai kumpulan objek yang berisi data dan operasi yang diberlakukan terhadapnya. Metodologi berorientasi objek merupakan suatu cara bagaimana sistem perangkat lunak dibangun melalui pendekatan objek secara sistematis[18]. Berikut ini adalah konsep dasar yang harus dipahami tentang metodologi berorientasi objek:

1. Kelas (*class*)  
*Class* adalah *prototype*, atau *blueprint*, atau rancangan yang mendefinisikan variabel dan metode-metode pada seluruh objek tertentu. *Class* berfungsi untuk menampung isi dari program yang akan dijalankan, didalamnya berisi atribut / tipe data dan method untuk menjalankan suatu program.
2. Objek (*object*)  
*Objek* adalah *instance* dari *class*. Jika *class* secara umum mempresentasikan (templates) sebuah objek, sebuah *instance* adalah representasi nyata dari *class* itu sendiri.
3. Metode (*method*)  
Metode merupakan suatu operasi berupa fungsi-fungsi yang dapat dikerjakan oleh suatu objek. Method didefinisikan pada *class* akan tetapi dipanggil melalui objek.

4. Atribut (*attribute*)  
Atribut merupakan nilai data yang terdapat pada suatu objek yang berasal dari *class*. *Attribute* mempresentasikan karakteristik dari suatu objek.
5. Abstraksi (*abstract*)  
Prinsip untuk merepresentasikan dunia nyata yang kompleks menjadi satu bentuk model yang sederhana dengan mengabaikan aspek-aspek lain yang tidak sesuai dengan permasalahan.
6. Enkapsulasi (*encapsulation*)  
Pembungkusan *attribute* data dan layanan (operasi-operasi) yang dipunyai objek untuk menyembunyikan implementasi dan objek sehingga objek lain tidak mengetahui cara kerjanya.
7. Pewarisan (*inheritance*)  
Mekanisme yang memungkinkan suatu objek mewarisi sebagian atas seluruh definisi objek lain sebagai bagian dari dirinya.
8. Antarmuka (*interface*)  
Antarmuka atau *interface* sangat mirip dengan kelas, tapi tanpa atribut kelas dan memiliki metode yang dideklarasikan tanpa isi. Deklarasi metode pada sebuah *interface* dapat diimplementasikan oleh kelas lain.

Keunggulan yang dihasilkan pada PBO diantaranya:

1. Meningkatkan produktivitas  
Karena kelas dan objek yang ditemukan dalam suatu masalah masih dapat dipakai diulang kembali untuk masalah lainnya yang melibatkan objek tersebut (*reusable*).
2. Kecepatan pengembangan  
Karena sistem yang dibangun dengan baik dan benar pada saat analisis dan perancangan akan menyebabkan berkurangnya kesalahan pada saat pengkodean.
3. Kemudahan pemeliharaan  
Karena dengan model objek, pola-pola yang cenderung tetap dan stabil dapat dipisahkan dan pola-pola mungkin sering berubahubah.
4. Adanya konsistensi  
Karena sifat pewarisan dan penggunaan notasi yang sama pada saat analisis, perancangan maupun pengkodean.
5. Meningkatkan kualitas perangkat lunak  
Karena pendekatan pengembangan lebih dekat dengan dunia nyata dan adanya konsistensi pada saat pengembangannya, perangkat

lunak yang dihasilkan akan mampu memenuhi kebutuhan pemakai serta mempunyai sedikit kesalahan

### **2.2.8 Framework**

*Framework* adalah suatu struktur yang didalamnya terdapat *class-class*, pustaka-pustaka dan infrastruktur *run-time* untuk membuat *web*, dengan menggunakan *framework* memungkinkan untuk membuat *web* dengan mudah dan cepat. *Framework* saat ini tersedia berbagai macam untuk mengembangkan *web*, salah satunya yaitu *CodeIgniter*. *CodeIgniter* (CI) merupakan *framework* PHP yang dibuat berdasarkan *Model-View-Controller* (MVC). CI memiliki *library* yang lengkap untuk mengerjakan operasi-operasi yang umum dibutuhkan oleh aplikasi berbasis *web* misalnya mengakses database, memvalidasi form sehingga sistem yang dikembangkan mudah. CI menjadi satu-satunya *framework* dengan dokumentasi yang lengkap dan jelas. *Source code* CI yang dilengkapi dengan *comment* di dalamnya sehingga lebih memperjelas fungsi sebuah kode program dan CI yang dihasilkan sangat Bersih (*clean*) dan *Search Engine Friendly* (SEF).

### **2.2.9 DBMS (Data Base Management System)(data base)**

Pada dasarnya arti kata *data base* merupakan gabungan dari kedua kata yaitu *data* dan *base*, *data* merupakan sekumpulan catatan maupun gambaran yang sesuai dengan fakta, sedangkan *base* atau dalam bahasa Indonesia disebut sebagai basis merupakan suatu wadah seperti gudang sebagai tempat berkumpulnya data-data. Jadi *data base* merupakan sekumpulan data-data yang penting dalam suatu informasi yang berfungsi untuk menyediakan informasi/data bagi para penggunanya, sedangkan DBMS (*Data base Management System*) merupakan sistem *software* yang diciptakan oleh user berupa data-data agar dapat dipelihara, dikontrol, serta dapat diakses secara efisien.

DBMS adalah jembatan penghubung antara *data base* dengan program aplikasi, beberapa tujuan dan DBMS pada jaringan komputer adalah agar tidak terjadinya kekacauan dalam hal pengelolaan data dengan skala yang besar, selain itu agar data yang diakses lebih cepat & mudah serta terjamin keamanannya [15].

#### **2.2.9.1 Basis data**

*Basis data* adalah kumpulan data yang saling berelasi. *Data* merupakan fakta mengenai objek, orang, dan lain-lain. *Data* dinyatakan

dengan nilai (angka, deretan karakter, atau simbol) [16]. *Basis data* dikelola dengan cara menuliskan sebuah kode perintah berupa *query* SQL yang harus dimasukan sesuai kebutuhan dan hal tersebut terbagi menjadi 2 yaitu, DDL (*Data Definition Language*) dan DML (*Data Manipulation Language*). Berikut dibawah ini merupakan pengertian DDL dan DML:

1. DDL (*Data Definition Languages*)

DDL digunakan sebagai perintah yang berhubungan dengan pendefinisian dari suatu struktur database. Adapun perintah-perintah dasar pada DDL:

a. *Create*

Perintah dalam DDL untuk membuat database serta tabel. Contoh perintah *create*, “*create table tb\_daftarpk1, email VARCHAR(160) NOT NULL, nis VARCHAR(35) NOT NULL, sekolah VARCHAR(160), jurusan (VARCHAR 100), tgl\_mulai DATE, tgl\_selesai DATE, status VARCHAR(20), catatan VARCHAR(255) )*”.

b. *Rename*

Perintah yang digunakan untuk mengganti nama tabel. Contoh perintah *rename*, “*rename table tb\_daftarpk1 to tb\_daftarpk1\_sekda;*”

c. *Alter*

Perintah yang digunakan untuk mengedit atau mengubah. Contoh perintah *alter*, “*alter table tb\_daftarpk1 drop column email;*”

d. *Drop*

Perintah yang digunakan untuk menghapus tabel maupun *database*. Contoh perintah *drop*, “*drop table tb\_daftarpk1;*”

2. DML (*Data Management Languages*)

DML merupakan perintah yang digunakan untuk melakukan pengelolaan pada database seperti membuat baru, menghapus, menambah dan menampilkan. Berikut ini adalah perintah pada DML :

a. *Insert*

Perintah yang berfungsi untuk memasukkan data ke dalam *database*. Contoh perintah *insert*, “*insert into tb\_persyaratan (id\_persyaratan,id\_daftarpk1, id\_dokumen,tipe\_file) VALUES ("0243", "54", "11", "dokumen");*”.

- b. *Update*  
Perintah yang digunakan untuk pembaruan data. Contoh perintah dari *update*, “*Update tb\_daftarplk Set id\_daftarplk =”200202074” Where jurusan= Teknik Informatika;*”.
- c. *Select*  
Perintah yang berfungsi untuk menampilkan data. Contoh perintah *select*, “*Select \* From tb\_daftarplk;*”.
- d. *Delete*  
Perintah yang digunakan untuk menghapus data. Contoh perintah dari *delete*, “*Delete From tb\_daftarplk Where jurusan =“ Pendidikan Pancasila;*”.