BAB II
TINJAUAN PUSTAKA DAN LANDASAN TEORI

2.1 Tinjauan Pustaka

Penelitian yang dilakukan oleh Anton Sujarwo, Adika May Sari, Rina Lestari dan Desri Yani di Universitas BSI pada tahun 2020 dengan judul “Sistem Informasi Pengajuan Klaim Asuransi Kendaraan Berbasis Web Menggunakan UML”. Tujuan dari penelitian ini adalah membuat sistem informasi Pengajuan Klaim Kendaraan yang efektif dan efisien dengan terkomputerisasinya sebuah sistem menggunakan berbasis *Website*, untuk membantu memperlancar pengelolaan informasi Pengajuan Klaim Kendaraan.Untuk mengembangkan sistem informasi pengajuan Klaim Kendaraan yang menggunakan metode *waterfall*. Hasil pengujian menunjukan bahwa sistem informasi mampun melakukan proses Pengajuan Klaim Kendaraan secara online dapat efektif dan efisien[3].

Penelitian yang lain juga dilakukan oleh Akhmad Syarifudin dan Nur Ani dengan judul "Rancangan Sistem Informasi Pengajuan dan Pelaporan Tunjangan Kinerja Kementerian Keuangan Menggunakan Metode *Prototype*" pada tahun 2019. Penelitian ini dilakukan untuk membantu Biro Perencanaan dan Keuangan Kementerian Keuangan dalam mengelola tunjangan kinerja. Proses Pengajuan dan pelaporan pembayaran tunjangan kinerja selama ini masih menggunakan cara manual sehingga membutuhkan waktu yang cukup lama dari satker sampai ke kementerian. Metode yang digunakan dalam pengembangan sistem adalah *prototyping*. Selain mempermudah proses pengajuan dan pelaporan tunjangan kinerja, aplikasi juga dirancang agar kementerian dapat memonitor progres pengajuan dan pelaporan pada berbagai tingkatan baik unit eselon I, kanwil maupun satker[4].

Pada tahun 2022 juga terdapat penelitian dengan judul "Sistem Informasi Pengajuan Nota Dinas (Studi di Dinas Pendidikan dan Kebudayaan Kabupaten Kuningan)" yang dilakukan oleh Aji Permana dari Fakultas Ilmu Komputer Universitas Kuningan. Pada penelitian ini dibahas mengenai proses pengajuan anggaran, realisasi, dan pelaporan tidak ada kesesuian. Penelitian ini bertujuan membuat rumusan serta rancangan sistem informasi membuat lapiran melalui sistem informasi

pengajuan nota dinas. Metode pengembangan yang dilakukan pada penelitian ini adalah metode *waterfall*. Penelitian ini berhasil membantu setiap bidang dikarenakan terdapat firut history yang terus menyimpan data. Selain itu pimpinan berhasil melihat realisasi dana yang ada[5].

Dengan melihat kajian penelitian sebelumnnya, penulis mengangkat judul "Sistem Infomasi *E-vouching* dengan Notifikasi *WhatsApp* berbasis *Website*" dimana terdapat perbedaan penelitian ini dengan kajian penelitian sebelumnya antara lain: Sistem yang dibangun ini menerapkan fitur *WhatsApp Gateway* yang mana sistem dapat mengirim notifikasi kepada Manajer dan Bendahara untuk memberi persetujuan ketika pengguna melakukan permohonan dana. Sistem juga dapat menyajikan history transaksi permohonan dana yang telah di setujui dan Bendahara dapat membuat laporan pada sistem.

2.2 Landasan Teori

2.2.1 Sistem Informasi

Sistem informasi adalah kumpulan elemen yang saling terhubung membentuk satu kesatuan untuk mengintegrasikan data, pemrosesan dan penyimpanan serta informasi distribusi[6]. Sistem Informasi juga dapat diartikan sebagai kumpulan elemen-elemen yang saling terkait untuk mencapai sebuah tujuan, Sistem mengelola data kemudian menghasilkan informasi yang berguna bagi *user*.

2.2.2 Rekayasa Perangkat Lunak

Rekayasa Perangkat Lunak singkatan dari RPL adalah suatu disiplin ilmu yang mana dibuat untuk kebutuhan membangun *software* yang bebas dari kesalahan, pengiriman yang sesuai dengan waktunya dan memenuhi keinginan penggunanya[7]. Dapat dikatakan RPL merupakan sebuah disiplin ilmu yang membahas semua bagian produksi peranngkat lunak, diawali dari tahap awal yaitu *communication requirements capturing* (analisa kebutuhan *user*), *specification* (menentukan spesifikasi dari kebutuhan *user*), desain, coding, testing hingga pemeliharaan sistem setelah digunakan.

A. Metode Pengembangan Sistem

Metode pengembangan sistem pada sistem ini yaitu metode *prototype*. Metode *Prtotype* merupakan proses pembuatan sistem yang bersifat berulang, memiliki perencanaan yang cepat, dimana terhadap

umpan balik yang memungkinkan terjadinya perulangan dan perbaikan sistem sampai sistem tersebut memenuhi kebutuhan dari pengguna. Berikut tahapan Metode *Prototype* menurut Pressman (2012) yaitu[8]:

1. Pengumpulan Kebutuhan
   Merupakan tahapan pengumpulan kebutuhan dengan melakukan observasi dan wawancara secara langsung dengan pihak Gerak Sedekah Cilacap untuk memperoleh data mengenai semua kebutuhan dan garis besar sistem yang akan dibangun.
2. Perancangan *Prototype*
   Setelah mengetahui kebutuhan, tahapan selanjutnya yaitu perancangan *Prototype*. Pada tahap pembuatan *prototype* yang dilakukan adalah pembuatan rancangan *Unified Modeling Language* (UML) yang terdiri atas *use case* diagram, *activity* diagram, serta *flowchart* sistem yang akan dibangun.
3. Evaluasi *Prototype*
   Merupakan tahap yang dilakukan oleh pengguna apakah *prototype* yang sudah dibuat sesuai dengan keinginan pengguna. Jika belum sesuai maka akan mengulangi Langkah 1 dan 2 kembali sesuai yang diinginkan dan dibutuhkan pengguna.
4. Pengkodean Sistem
   Dalam tahap ini *prototype* yang sudah disepakati diterjemahkan ke dalam bahasa pemrograman yang sesuai, pada tahapan ini peneliti akan menggunakan bahasa pemrograman PHP dan *database* MySQL.
5. Pengujian Sistem
   Setelah sistem sudah menjadi suatu *software* yang siap dipakai, harus dites atau di ujicoba dahulu sebelum digunakan yang bertujuan untuk mengetahui kesalahan yang ada pada aplikasi, memastikan fungi dari setiap fitur akan digunakan menghasilkan *output* yang diinginkan. Pengujian ini dilakukan dengan metode *blackbox testing.*
6. Evaluasi Sistem
   Pada tahap ini pengguna mengevaluasi apakah sistem yang sudah jadi sesuai dengan yang diharapkan. Jika sudah sesuai, maka akan ketahap selanjutnya yaitu penggunaan sistem.
7. Menggunakan Sistem *(Implementasi)*
   Tahap ini perangkat lunak yang telah diuji dan diterima pengguna siap untuk digunakan.

B. Metode Pengujian Sistem

Pengujian *black-box* adalah metode pengujian yang tidak memperhatikan implementasi internal dari sistem yang diuji, melainkan hanya menguji sistem berdasarkan fungsinya yang didefinisikan oleh spesifikasi[9]. Dalam metode ini, tester hanya mengetahui *input* dan *output* dari sistem yang diuji, tanpa mengetahui proses internal yang digunakan untuk menghasilkan *output* tersebut. Metode ini digunakan untuk menguji sistem yang kompleks atau sistem yang diimplementasikan oleh pihak ketiga.

Beberapa teknik pengujian *black box* yang umum digunakan di antaranya:

a. Uji coba masukan *(Input Testing)*
   Memasukkan berbagai jenis input ke sistem dan mengamati output yang dihasilkan.
b. Uji coba validasi *(Validation Testing)*
   Memastikan bahwa sistem dapat menangani input yang diharapkan dan tidak dapat menangani input yang tidak diharapkan.
c. Uji coba boundary *(Boundary Testing)*
   Mengidentifikasi dan menguji sistem pada batas-batas input yang valid.
d. Uji coba pengujian alur kerja *(Workflow Testing)*
   Memastikan bahwa sistem dapat menangani alur kerja yang ditentukan sesuai spesifikasi.
e. Uji coba pengujian kompatibilitas *(Compatibility Testing)*
   Memastikan bahwa sistem dapat berinteraksi dengan sistem lain yang sesuai dengan spesifikasi.
f. Uji coba pengujian performa *(Performance Testing)*
   Mengukur kinerja sistem dalam hal waktu respon, kapasitas dan skalabilitas.
g. Uji coba pengujian keamanan *(Security Testing)*
   Memastikan bahwa sistem tidak dapat diakses atau diubah oleh pihak yang tidak berwenang.
h. Uji coba pengujian regresi *(Regression Testing)*
   Memastikan bahwa perubahan yang dibuat pada sistem tidak

menyebabkan masalah pada bagian lain dari sistem.
Teknik pengujian yang digunakan akan tergantung pada tujuan pengujian dan spesifikasi sistem yang diuji.

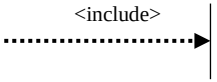
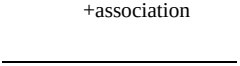
2.2.3 UML *(Unified Modelling Language)*

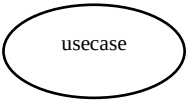
UML (*Unified Modelling Language*) merupakan analisa dan pemodelan desain sebuah pengembangan *software* penting untuk memastikan kualitas proses dan produk. UML sendiri juga sebagai alat perancang sistem pengembangan perangkat lunak yang berorientasi pada *object oriented (OO)*[13].

a. ***Use Case Diagram***

Use Case Diagram merupakan jenis diagram yang berfungsi untuk menjelaskan suatu interaksi antara satu atau lebih *actor* dengan sistem informasi yang akan dibangun dan dapat mendeskripsikan fungsi apa saja yang ada pasa sebuah sistem informasi[14].

Tabel 2.1 Tabel Use Case Diagram

No	Simbol	Fungsi
1		*Segala sesuatu yang berinteraksi dengan sistem aplikasi komputer. Jadi actor ini bisa berupa orang, perangkat keras atau mungkin juga obyek lain dalam sistem yang sama.*
2		*Menspesifikasikan bahwa prilaku* use case *merupakan bagian dari* use case *lain.*
3		*Menggambarkan navigasi antar* class, *berupa banyak obyek lain yang berhubungan dengan satu obyek, dan apakah*

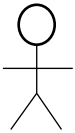
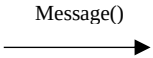
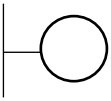
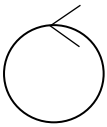
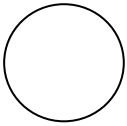
		suatu class *menjadi bagian dari* class *lainnya.*
4		*System Boundary* yaitu batasan sebuah sistem.
5		*Use case* menjelaskan urutan kegiatan yang dilakukan aktor dan sistem untuk mencapai suatu tujuan tertentu. Walaupun menjelaskan kegiatan, use case hanya menjelaskan apa yang dilakukan oleh aktor dan sistem, bukan bagaimana aktor dan sistem melakukan kegiatan.

b. ***Sequence Diagram***

Sequence Diagram adalah sebuah diagram yang digunakan dalam pemodelan objek untuk menggambarkan interaksi antar objek dalam suatu sistem. Sequence diagram menggambarkan alur waktu dari interaksi antar objek, dan menggambarkan bagaimana objek-objek tersebut berinteraksi satu sama lain dalam suatu proses[14].

Tabel 2.2 Simbol Sequence Diagram

No	Simbol	Fungsi

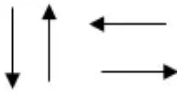
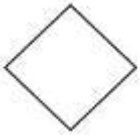
1		*Objek entity*, antarmuka yang saling berinteraksi
2		Digunakan untuk menggambarkan *user*/Pengguna
3		Spesifikasi dari komunikasi antar objek yang memuat informasi – informasi tentang aktifitas yang terjadi.
4		Digunakan untuk menggambarkan sebuah *form.*
5		Digunakan untuk menghubungkan *Boundary* dengan tabel
6		Digunakan untuk menggambarkan hubungan kegiatan yang akan dilakukan

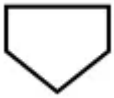
c. *Flowchart*

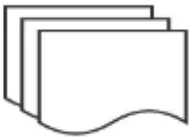
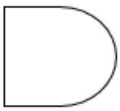
Flowchart adalah sebuah diagram alir yang digunakan untuk menggambarkan alur kerja suatu sistem atau proses. Flowchart menggambarkan langkah-langkah yang harus dilakukan dalam suatu proses, beserta kondisi-kondisi yang harus dipenuhi dan

keputusan-keputusan yang harus diambil. Berikut ini simbol – simbol *flowchart*[14]:

Tabel 2.3 Simbol Flowchart

No	Gambar Simbol	Nama	Keterangan
1.		*Terminal point*	Menunjukkan mulai atau berakhirnya suatu proses.
2.		*Flow direction*	Simbol yang digunakan untuk menghubungkan antar simbol, berfungsi juga untuk menunjukkan alur sebuah proses.
3.		Proses	Menunjukkan sebuah kegiatan yang dilakukan oleh komputer atau sistem.
4.		*Decision*	Simbol yang digunakan untuk memilih sebuah keputusan atau proses sesuai dengan kondisi yang ada.

5.		*Input output*	Menunjukkan sebuah *input* atau *output* yang ada tidak bergantung pada jenis peralatannya.
6.		*Predefined process*	Menunjukkan pelaksanaan suatu bagian prosedur, bagian prosedut yang terinformasi belum detail dan akan diperinci ditempat lain.
7.		*Connector (On-page)*	Menghubungkan suatu simbol pada satu halaman yang letaknya berjauhan.
8.		*Connector (Off-page)*	Menghubungkan simbol yang berada dalam halaman yang berbeda.
9.		*Preparation*	Simbol ini menunjukkan persiapan penyimpanan kedalam *storage*.

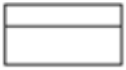
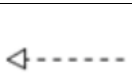
10.		*Manual input*	Menunjukkan proses *input* yang dilakukan secara *manual* menggunakan *online keyboard*.
11.		*Manual operation*	Menunjukkan proses/kegiatan yang dilakukan tanpa menggunakan komputer.
12.		*Document*	Menunjukkan *input* berupa dokumen dalam bentuk kertas atau *output* yang yang harus dicetak.
13.		*Multiple document*	Sama seperti simbol *document* hanya saja dokumen yang digunakan lebih dari satu.
14.		*Display*	Simbol yang menunjukkan adanya penggunaan peralatan *output*.

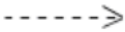
15.		*Delay*	Menunjukkan bahwa adanya proses *delay*.

d. *Class Diagram*

Class diagram adalah sebuah diagram yang digunakan dalam pemodelan objek untuk menggambarkan struktur kelas-kelas dalam suatu sistem. Diagram ini menggambarkan hubungan antar kelas, atribut dan operasi yang dimiliki oleh setiap kelas, serta hubungan antar kelas.

Tabel 2.4 Simbol *Class Diagram*

No	Gambar Simbol	Nama	Keterangan
1.		*Generilaziation*	Relasi antarkelas dengan makna generalisasi-spesialisasi (umum-khusus).
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.		*Realization*	Operasi yang dilakukan oleh

			suatu objek.
5.		*Dependency*	Relasi antarkelas dengan makna kebergantungan antarkelas.
6.		*Association*	Diagram yang menghubungkan antara objek satu dengan objek lainnya.

2.2.4 *Database*

Database atau basis data adalah kumpulan dari berbagai data yang tersimpan dan tersusun dengan teratur di dalam komputer agar dapat dikelola dengan suatu perangkat lunak sehingga menghasilkan informasi yang berguna bagi penggunanya. Dengan adanya *database* pengguna mudah mengelompokan data dan terhindar adanya duplikat data[12].

Database 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[13]:

1. DDL

DDL digunakan untuk pengoperasian skema struktur pada *database*, perintah utama yang dapat digunakan pada DDL dianataranya, *create*, *rename*, *alter*, dan *drop*.

a. *Create*

Perintah untuk membuat *database* dan *table.* Contoh perintah, "CREATE TABLE Buku_Perpus(id INTEGER PRIMARY KEY, kode_buku VARCHAR(20) NULL, judul_buku VARCHAR(225) NOT NULL, tanggal_terbit DATE NULL)".

b. *Rename*
Perintah yang digunakan untk mengganti nama *table.* Contoh perintah**, "**RENAME TABLE Jurnal TO Jurnalku**;**"

c. *Alter*
Perintah yang digunakan untuk merubah atau mengedit. Contoh perintah,"Alter TABLE Buku ADD Penulis Varchar(100); Alter TABLE Buku Drop Column Judul_buku;".

d. *Drop*
Drop digunakan untuk menghapus *table* maupun *database.* Contoh perintah "Drop Table Buku;".

2. DML

DML merupakan perintah yang dipakai untuk melakukan pengelolaan pada *database* seperti menambah, menghapus, membuat baru, dan menampilkan. Perintah utama pada DML diantranya adalah *update, delete, insert,* dan *select.*

a. *Select*
Perintah yang berfungsi untuk menampilkan data yang telah dimasukan. Contoh perintah, "SELECT * FROM Buku;"

b. *Insert*
Perintah untuk memasukan data ke dalam *database.* Contoh perintahnya, "INSERT INTO Buku (id, kode_buku, judul_buku, tanggal_terbit) values(1124, "KoD2″, "Desainer", "2013-01-11″);"

c. *Update*
Perintah yang dilakukan untuk pembaruan data. Contoh perintahnya, "UPDATE Buku SET Judul_buku = "Programmer" WHERE id = 1123;"

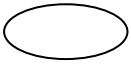
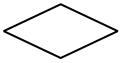
d. *Delete*
Perintah untuk menghapus data. Contoh perintah, "DELETE FROM Buku WHERE judul="Programmer";"

3. *Entity Relationship Diagram (ERD)*

ERD adalah suatu rancangan atau bentuk hubungan suatu kegiatan di dalam sistem yang berkaitan langsung dan mempunyai fungsi di dalam proses tersebut. ERD adalah suatu pemodelan dari basis data relasional yang didasarkan atas persepsi di dalam dunia nyata, dunia ini senantiasa terdiri dari sekumpulan objek yang saling berhubungan antara satu dengan yang lainnya. Suatu objek disebut *entity* dan hubungan yang

dimilikinya disebut *relationship*. Suatu *entity* bersifat unik dan memiliki atribut sebagai pembeda dengan *entity* lainnya. Berikut merupakan simbol-simbol dari ERD:

Tabel 2.5 Simbol 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.	Link		Garis sebagai penghubung antara himpunan, relasi, dan himpunan entitas dengan atributnya

ERD memiliki derajat relasi atau biasa disebut kardinalitas. Kardinalitas menjelaskan batasan jumlah keterhubungan satu entity dengan entity lainnya.

Tabel 2.6 Macam-Macam Kardinalitas

No	Nama	Simbol	Keterangan
1	Relasi Satu ke Satu (*One to One*)		Relasi yang menunjukkan bahwa setiap himpunan entitas berhubungan dengan tepat satu himpunan entitas lainnya

2	Relasi Satu ke Banyak (*One to Many*)		Relasi yang menunjukkan bahwa hubungan antara entitas pertama dengan entitas kedua adalah satu banding banyak, begitu pula sebaliknya
3	Relasi Banyak ke Banyak (*Banyak to Many*)		Relasi yang menunjukkan bahwa setiap himpunan entitas boleh berhubungan dengan banyak himpunan entitas lainnya dan sebaikanya

2.2.5 *Vouching*

Vouching adalah sebuah kegiatan melakukan pemeriksaan transaksi bisnis, dengan memeriksa dokumen, catatan, atau bukti-bukti lainnya yang mempunyai cukup keabsahan untuk memenuhi pertimbangan auditor bahwa transaksi tersebut telah benar, telah diotorisasi secara tepat, dan telah dicatat dalam pembukuan dengan benar[19].

Tujuan dilakukannya *vouching* untuk memastikan bahwa:

1) Bukti tersebut telah disetujui oleh pejabat yang berwenang dan tekait.
2) Bukti tersebut sesuai dengan tujuannya.
3) Jumlah yang tertera didalam bukti adalah benar telah sesuai dengan transaksinya.
4) Pencatatan dilakukan secara benar.

Halaman ini sengaja dikosongkan