

**PENGEMBANGAN APLIKASI PEMBAYARAN SISWA
PADA YAYASAN PERGURUAN ISLAM AS SA'ADAH
PONDOK KELAPA**

Naskah Publikasi Jurnal



Diajukan oleh:

TRISNA YUDHA
5235107398

**PROGRAM STUDI PENDIDIKAN TEKNIK INFORMATIKA DAN KOMPUTER
JURUSAN TEKNIK ELEKTRO - FAKULTAS TEKNIK
UNIVERSITAS NEGERI JAKARTA
[2016]**

NASKAH PUBLIKASI JURNAL

**PENGEMBANGAN APLIKASI PEMBAYARAN SISWA
PADA YAYASAN PERGURUAN ISLAM AS SA'ADAH
PONDOK KELAPA**

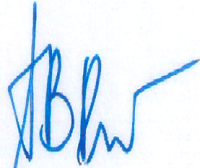
yang diajukan oleh :

TRISNA YUDHA

5235107398

Telah disetujui oleh :

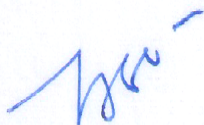
Pembimbing 1



Bambang Prasetya Adhi, S.Pd., M.Kom
NIP. 198302252014041001

Tanggal 26 Januari 2016

Pembimbing 2



Drs. Bachren Zaini, M.Pd
NIP.195501101986021001

Tanggal 27 Januari 2016

PENGEMBANGAN APLIKASI PEMBAYARAN SISWA PADA YAYASAN PERGURUAN ISLAM AS SA'ADAH PONDOK KELAPA

Trisna Yudha¹, Bambang Prasetya Adhi, S.Pd., M.Kom², Drs. Bachren Zaini, M.Pd³

¹ Mahasiswa Prodi Pendidikan Teknik Informatika dan Komputer, Teknik Elektro, FT – UNJ

^{2,3} Dosen Prodi Pendidikan Teknik Informatika dan Komputer, Teknik Elektro, FT – UNJ

¹trisna1976@gmail.com, ²bambangpadhi@unj.ac.id, ³bzaini@ft.unj.ac.id

Abstrak

Semakin rumitnya sistem pengolahan data pada bagian pelayanan keuangan, serta untuk menghindari kesalahan dalam mengkalkulasikan data. Yayasan Perguruan Islam As Sa'adah Pondok Kelapa telah berupaya melakukan pengolahan data secara terkomputerisasi tetapi belum maksimal. Disebabkan sistem pembayaran siswa masih secara konvensional menggunakan *spreadsheet* dan manual (buku). Dikarenakan pembuatan laporan yang masih belum akurat karena data dalam pencatatan dan input data sering terjadi kesalahan, proses pembuatan laporan memakan waktu yang cukup lama, dan tidak ada informasi keterangan apakah siswa tersebut sudah lunas atau belum dalam pembayaran biaya SPP, PSB, Uang Pangkal, UTS/UAS. Dalam penelitian ini dikembangkan Aplikasi Pembayaran Siswa yang dapat mempermudah petugas keuangan dalam proses pencatatan dan pelaporan pembayaran siswa. Sehingga diharapkan peningkatan kinerja dan pelayanan petugas keuangan dalam menyelesaikan masalah pembayaran siswa dapat terwujud. Penelitian ini menggunakan metode eksperimen. Proses penyimpanan pada aplikasi menggunakan model basis data. Aplikasi ini dikembangkan dengan menggunakan metode *prototype*, menggunakan bahasa pemrograman PHP dan perangkat lunak basis data MySQL dengan XAMPP sebagai *local server*. Pengujian kelayakan aplikasi diuji oleh user dengan menggunakan metode *User Acceptance Test (UAT)*. Dari hasil pengujian dapat disimpulkan bahwa Aplikasi Pembayaran Siswa berfungsi dengan baik dan dapat dimanfaatkan pada Yayasan Perguruan Islam As Sa'adah.,

Kata kunci : aplikasi, pembayaran siswa, *prototype*, *User Acceptance Test (UAT)*.

1. Pendahuluan

Dunia pendidikan merupakan sebuah bagian yang tidak dapat dipisahkan dari kehidupan bermasyarakat. Pendidikan identik dengan proses belajar mengajar, karena merupakan kegiatan utama dalam dunia pendidikan. Disamping kegiatan belajar mengajar ada banyak faktor pendukung dalam kegiatan belajar mengajar. Salah satunya adalah kegiatan berkaitan dengan bagian administrasi. Tanpa dukungan kinerja bagian administrasi kegiatan mengajar tentunya tidak maksimal untuk menciptakan suasana akademik yang stabil. Bagian administrasi merupakan bagian yang melakukan kegiatan rutin misalnya: menulis, menghitung serta mengolah data pada suatu instansi pendidikan. Pengembangan Aplikasi sudah sering kita jumpai hampir di setiap perusahaan ataupun instansi pendidikan sebagai sarana pendukung. Aplikasi dikembangkan dan dipergunakan untuk melakukan pekerjaan-pekerjaan tertentu. Diantaranya adalah Aplikasi Pembayaran Siswa yang sangat membantu dan mempermudah petugas keuangan dalam proses pencatatan dan pelaporan pembayaran siswa. Teknologi komputer sangat mendukung dalam hal pengolahan data untuk membuat suatu aplikasi yang digunakan agar memberikan suatu hasil kerja yang

maksimal. Pengguna komputer dalam sebuah aplikasi tidak terlepas dari pengguna *hardware* dan *software* serta *brainware* yang handal dalam menjalankan sistem agar bekerja optimal dan sesuai dengan kebutuhan. Kebutuhan aplikasi berbasis web semakin meningkat. Web menjadi teknologi pilihan bagi perusahaan maupun personal untuk menampilkan informasi-informasi tentang profil, produk, layanan atau informasi lain yang bermanfaat bagi publik. Salah satu adalah aplikasi pembayaran siswa berbasis web. Siswa dapat mengetahui jumlah tunggakan yang belum dibayar dan sekolah dapat mengetahui siswa yang sudah membayar maupun yang belum membayar secara jelas. Jumlah keseluruhan siswa yang ada di lembaga pendidikan formal pada Yayasan Perguruan Islam As Sa'adah Pondok Kelapa kurang lebih saat ini 846 orang terdiri dari TK, SD, SMP, dan SMK. Proses pengolahan data administrasi sekolah sepenuhnya dilakukan oleh tata usaha. Mengingat semakin meningkatnya jumlah siswa dan siswi di lembaga pendidikan formal Yayasan Perguruan Islam As Sa'adah Pondok Kelapa dan semakin rumitnya sistem pengolahan data pada bagian pelayanan keuangan, serta untuk menghindari kesalahan dalam mengkalkulasikan data tersebut, maka Yayasan Perguruan Islam As Sa'adah Pondok Kelapa telah

berupaya melakukan pengolahan datanya secara terkomputerisasi tetapi belum maksimal, yaitu sistem pembayaran siswa masih menggunakan *spreadsheet* dan manual (buku), pembuatan laporan pembayaran siswa masih belum akurat karena data dalam pencatatan dan dalam input data sering terjadi kesalahan dan pembuatan laporan pembayaran siswa masih belum tepat waktu karena data – data yang sudah masuk harus menunggu sampai data tersebut terkumpul sehingga prosesnya memakan waktu yang cukup lama, sistem pembayaran siswa yang berlaku pada saat ini masih belum optimal, masih ada kekurangan yaitu tidak adanya informasi keterangan apakah siswa tersebut sudah lunas atau belum dalam pembayaran biaya SPP, PSB, Uang Pangkal, UTS/UAS. SDLC (*Systems Development Life Cycle*) siklus hidup pengembangan sistem atau *Systems Life Cycle* (siklus Hidup Sistem) dalam rekayasa perangkat lunak adalah proses pembuatan dan perubahan sistem serta model dan metodologi yang digunakan untuk mengembangkan sistem-sistem tersebut. SDLC mendasari berbagai jenis metodologi pengembangan perangkat lunak. Metodologi-metodologi ini membentuk suatu kerangka kerja untuk perencanaan dan pengendalian pembuatan sistem informasi, yaitu proses pengembangan perangkat lunak. Terdapat tiga jenis SDLC yang paling banyak digunakan, yakni : siklus hidup sistem tradisional, siklus hidup menggunakan *prototyping* dan siklus hidup sistem orientasi objek. Dengan pertimbangan diatas, maka penulis mengadakan penelitian untuk mengembangkan Aplikasi Pembayaran Siswa yang sistematis mengacu pada optimalisasi penggunaan komputer yang dapat mempermudah petugas keuangan dalam proses pencatatan dan pelaporan pembayaran siswa. Untuk itu penulis mengambil judul “Pengembangan Aplikasi Pembayaran Siswa pada Yayasan Perguruan Islam As Sa’adah Pondok Kelapa” dan untuk mengefisienkan waktu dalam perancangan dan desain serta meminimalisir keadaan *miss* komunikasi antara *software developer* dan *customer* maka pada pengembangan perangkat lunak ini menggunakan metode *prototype* dalam SDLC.

2. Dasar Teori

2.1 Perangkat Lunak

Perangkat lunak atau sering disebut *software* adalah 1) Perintah (program komputer) yang apabila dijalankan menghasilkan fungsi dan unjuk kerja yang diinginkan. 2) Struktur data yang memungkinkan program memanipulasi informasi secara proposional. 3) Dokumen yang menggambarkan operasi dan penggunaan dari perangkat lunak (Pressman, 2010: 6). Perangkat lunak tidak hanya program, tetapi juga semua yang terkait dokumen dan data konfigurasi yang diperlukan untuk membuat program-program beroperasi dengan benar

(Sommerville, 2001: 5). Perangkat lunak lebih mengacu kepada *logical* daripada *physical system* element. Oleh karena itu, perangkat lunak mempunyai karakteristik yang membedakan dengan perangkat keras yaitu : 1. Perangkat lunak dikembangkan atau direkayasa bukan diciptakan. 2. Perangkat lunak tidak akan habis atau hilang. 3. Meskipun industri bergerak ke arah komponen berbasis konstruksi tetapi sebagian besar perangkat lunak tetap dibangun atau dikembangkan (Pressman, 2010: 6-10).

2.1.1 Rekayasa perangkat lunak

Rekayasa Perangkat lunak adalah pembuatan dan penggunaan prinsip-prinsip keahlian teknik untuk mendapatkan perangkat lunak yang ekonomis handal dan bekerja secara efisien pada mesin yang sesungguhnya (Fritz Bauer, 1969). Definisi Rekayasa perangkat lunak yang lebih komprehensif adalah (1) Aplikasi dari sebuah pendekatan kuantitatif, disiplin, dan sistematis kepada pengembangan, operasi, dan pemeliharaan perangkat lunak; yaitu aplikasi dari rekayasa perangkat lunak. (2) Studi tentang pendekatan-pendekatan seperti pada (1) (IEEE, 1993). Rekayasa perangkat lunak adalah prinsip tentang perkerayaan yang berhubungan dengan semua aspek dari pembuatan perangkat lunak dari tahap awal spesifikasi sistem sampai perawatan sistem setelah memasuki tahap penggunaan (Sommerville, 2001: 6).

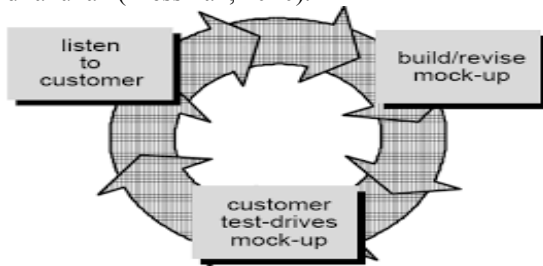
2.1.2 Metodologi Pengembangan Perangkat Lunak

Metodologi pengembangan Perangkat Lunak adalah suatu metode untuk mengembangkan perangkat lunak. Metode-metode pengembangan perangkat lunak memberikan teknik untuk membangun perangkat lunak. Metode-metode itu menyangkut serangkaian tugas yang luas yang menyangkut analisis kebutuhan, konstruksi program, desain, pengujian, dan pemeliharaan. Rekayasa perangkat lunak mengandalkan pada serangkaian prinsip dasar yang mengatur setiap area teknologi dan menyangkut aktivitas pemodelan serta teknik-teknik deskriptif yang lain.

2.1.3 Model Prototype

Metode Pengembangan Perangkat Lunak Prototype merupakan salah satu metode pengembangan perangkat lunak yang bersifat iteratif. Metode ini cocok digunakan pada saat stakeholder mendefinisikan beberapa tujuan umum dari perangkat lunak, tetapi tidak mendefinisikan kebutuhan rinci untuk fungsi dan fiturnya. Metode ini dimulai dengan komunikasi. Pengembang aplikasi bertemu dengan stakeholder untuk mendefinisikan keseluruhan tujuan dari perangkat

lunak dan mendefinisikan kebutuhan yang diketahui. Langkah selanjutnya adalah membuat perencanaan secara cepat dilanjutkan pemodelan yang berfokus pada aspek dari perangkat yang tampak pada stakeholder, contohnya adalah rancangan antar muka. Setelah selesai pemodelan, pembuatan prototype dilakukan, dipaparkan, dan dievaluasi oleh stakeholder untuk mengetahui kebutuhan selanjutnya. Pengulangan terjadi sampai prototype memuaskan kebutuhan berbagai stakeholder dan pada saat bersamaan memungkinkan pengembang aplikasi untuk memahami lebih baik apa yang perlu dilakukan (Pressman, 2010).



Gambar 2.1. The Prototyping Paradigm

Keuntungan dari prototype:

1. User dapat mempertimbangkan sedikit perubahan selama masih bentuk prototype.
2. Memberikan hasil yang lebih akurat dari pada perkiraan sebelumnya, karena fungsi yang diinginkan dan kerumitannya sudah dapat diketahui dengan baik.
3. User merasa puas. Pertama, user dapat mengenal melalui komputer. Dengan melakukan prototype (dengan analisis yang sudah ada), user belajar mengenai komputer dan aplikasi yang akan dibutuhkan untuknya. Kedua, user terlibat langsung dari awal dan memotivasi semangat untuk mendukung analisis selama proyek berlangsung.

2.1.4 Aplikasi

Aplikasi berasal dari kata application yang artinya penerapan, lamaran, penggunaan. Secara istilah aplikasi adalah program siap pakai yang dibuat untuk melaksanakan suatu fungsi bagi pengguna atau aplikasi yang lain dan dapat digunakan oleh sasaran yang di tuju (Pakpahan, 2015). Aplikasi adalah penggunaan dalam suatu komputer, intruksi (*instruction*) atau pernyataan (*statement*) yang disusun sedemikian rupa sehingga komputer dapat memproses input menjadi output (Jogiyanto, 1999). Menurut Kamus Besar Bahasa Indonesia, aplikasi adalah penerapan dari rancangan sistem untuk mengolah data yang menggunakan aturan atau ketentuan bahasa pemrograman tertentu. Aplikasi adalah suatu program komputer yang dibuat untuk mengerjakan dan melaksanakan tugas khusus dari pengguna (Kamus Besar, 1998:52). Menurut Ibis (1999) aplikasi adalah alat bantu untuk mempermudah dan mempercepat proses pekerjaan

dan bukan merupakan beban bagi penggunanya. Menurut Santoso (2000) aplikasi adalah suatu kelompok *file* (*form, class, report*) yang bertujuan untuk melakukan aktivitas tertentu yang saling terkait, misalnya aplikasi *payroll*, aplikasi *fixed asset*. Menurut Hendrayudi (2004) aplikasi adalah kumpulan perintah program yang dibuat untuk melakukan pekerjaan-pekerjaan tertentu.

2.1.5 Unified Modeling Language (UML)

Pada tahun 1994, Grady Booch dan James Rumbaugh sepakat menggunakan metode pengembangan berorientasi objek dengan tujuan membuat proses standar tunggal untuk mengembangkan sistem berorientasi objek. Ivar Jacobson bergabung pada tahun 1995, dan mereka bertiga membuat suatu bahasa pemodelan objek standar sebagai ganti dari pendekatan/metode berorientasi objek standar, yaitu *Unified Model Language* (UML) yang dirilis pada tahun 1997. Pemecahan masalah utama dari *Object Oriented* biasanya dengan penggambaran dalam bentuk model. Model abstrak (semu) merupakan gambaran detail dari inti masalah yang ada, umumnya sama seperti refleksi dari problem yang ada pada kenyataan. Beberapa modeling tool yang dipakai adalah bagian dari dasar UML, kependekan dari *United Modeling Language*. Menurut Whitten, dan Bentley (2007) UML menawarkan diagram-diagram dengan perspektif yang berbeda untuk memodelkan suatu sistem. UML terdiri atas beberapa diagram, yaitu : *Diagram Use Case, Diagram Class, Diagram Package, Diagram Sequence, Diagram Collaboration, Diagram StateChart, Diagram Activity, Diagram Deployment*. Semakin kompleks bentuk sistem yang akan dibuat, maka semakin sulit komunikasi antara orang-orang yang saling terkait dalam pembuatan dan pengembangan software yang akan dibuat. Pada masa lalu, UML mempunyai peranan sebagai *software blueprint* (gambaran) *language* untuk analisis sistem, *designer*, dan *programmer*. Sedangkan pada saat ini, merupakan bagian dari *software trade* (bisnis *software*). UML memberikan jalur komunikasi dari sistem analisis kemudian *designer*, lalu *programmer* mengenali rancangan *software* yang akan dikerjakan. Tahap pertama, pembentukan model. Model adalah gambaran abstrak dari suatu dasar masalah. Dan dunia nyata atau tempat dimana masalah itu timbul bisa disebut dengan domain. Model mengandung obyek-obyek yang beraktifitas dengan saling mengirimkan *messages* (pesan-pesan). Obyek mempunyai sesuatu yang diketahui (*atribut /attributes*) dan sesuatu yang dilakukan (*behaviors* atau *operations*). *Attributes* hanya berlaku dalam ruang lingkup obyek itu sendiri (*state*). Lalu "*blue print*" dari suatu obyek adalah *Classes* (kelas). Obyek merupakan bagian-bagian dari kelas.

2.1.5.1 Diagram Use Case

Diagram *Use Case* menggambarkan apa saja aktifitas yang dilakukan oleh suatu sistem dari sudut pandang pengamatan luar. yang menjadi persoalan itu apa yang dilakukan bukan bagaimana melakukannya. Diagram *Use Case* dekat kaitannya dengan kejadian-kejadian. Kejadian (*scenario*) merupakan contoh apa yang terjadi ketika seseorang berinteraksi dengan sistem. Diagram *Use Case* berguna dalam tiga hal yakni menjelaskan *requirements*, komunikasi dengan klien, membuat test dari kasus-kasus secara umum.

2.1.5.2 Diagram Class

Diagram *Class* memberikan pandangan secara luas dari suatu sistem dengan menunjukan kelas-kelasnya dan hubungan mereka. Diagram *Class* bersifat statis; menggambarkan hubungan apa yang terjadi bukan apa yang terjadi jika mereka berhubungan. Diagram *Class* mempunyai tiga macam *relationships* (hubungan), yaitu *association*, *aggregation*, dan *generalization*.

2.1.5.3 Diagram Sequence

Diagram *Class* dan diagram *Object* merupakan suatu gambaran model *statis*. Namun ada juga yang bersifat dinamis, seperti Diagram *Interaction*. Diagram *sequence* merupakan salah satu diagram *Interaction* yang menjelaskan bagaimana suatu operasi itu dilakukan; message (pesan) apa yang dikirim dan kapan pelaksanaannya. Diagram ini diatur berdasarkan waktu. Obyek-obyek yang berkaitan dengan proses berjalannya operasi diurutkan dari kiri ke kanan berdasarkan waktu terjadinya dalam pesan yang terurut.

2.1.6 Relation Database Manajemen System (RDBMS).

Kumpulan *file* yang saling berkaitan untuk program dan pengelolaannya disebut DBMS (Marlinda, 2004:6). DBMS adalah istilah yang penting dalam pengelolaan data. Untuk membuat dan mengelola data tersebut, dibutuhkan *software* yang diistilahkan DBMS (*Data Base Management System*). Tugas-tugas yang diemban *software* DBMS ini adalah membuat *database*, menampilkan data yang ada pada *database* tersebut, memodifikasi data pada *database* tersebut, menghasilkan laporan sesuai dengan data yang ada dalam *database*, dan mengamankan data dari pihak-pihak yang tidak berkepentingan (Swastika, 2006:36). Beberapa contoh *software* DBMS yang banyak beredar adalah *Oracle*, *Microsoft SQL Server*, *SyBase* dan *MySQL*. Hampir semua program DBMS merupakan RDBMS (*Relational DataBase Management System*), dimana data yang akan diorganisir dalam sekumpulan tabel

yang saling berelasi (berhubungan) (Swastika, 2006:36). Arsitektur Sistem Basis Data dibagi menjadi tiga tingkatan (Marlinda, 2004:9), yaitu: *Internal Level*, *External Level*, *Conceptual Level*. Kelemahan sistem pemrosesan berkas dapat diatasi dengan sistem *basis data*. RDBMS merupakan antar muka bagi pemakai dalam mengorganisasikan *database* yang disusun, pemakai dapat berinteraksi langsung dengan mudah dan praktis dengan menggunakan perintah-perintah yang sederhana yang di buat dalam suatu bahasa pemrograman.

2.1.7 Testing

Testing merupakan aktivitas atau proses memeriksa dan mengevaluasi sistem dengan tujuan untuk menemukan kesalahan pada sistem tersebut.

2.1.7.1 System Testing

System testing adalah pengujian yang dilakukan untuk memastikan bahwa sistem yang secara keseluruhan telah terintegrasi berfungsi dengan baik dan benar.

2.1.7.2 User Acceptance Testing

Acceptance Testing merupakan pengujian yang dilakukan oleh *end-user* dimana *user* tersebut adalah *staff* karyawan perusahaan yang langsung berinteraksi dengan sistem dan dilakukan verifikasi apakah fungsi yang ada telah berjalan sesuai dengan kebutuhan/fungsinya (Perry, 2006:70). Menurut Lewis (2009:134), setelah dilakukan *system testing*, *acceptance testing* menyatakan bahwa sistem *software* memenuhi persyaratan. *Acceptance testing* merupakan pengujian yang dilakukan oleh pengguna yang menggunakan teknik pengujian *black box* untuk menguji sistem terhadap spesifikasinya. Pengguna akhir bertanggung jawab untuk memastikan semua fungsionalitas yang relevan telah diuji. Menurut Black (2002:7), *acceptance testing* biasanya berusaha menunjukkan bahwa sistem telah memenuhi persyaratan-persyaratan tertentu. *Acceptance testing* mencakup data, *environment* dan skenario yang sama atau hampir sama pada saat *live* yang biasanya berfokus pada skenario penggunaan produk tertentu. Dari definisi di atas, *user acceptance testing* adalah pengujian yang dilakukan oleh pengguna dari sistem tersebut untuk memastikan fungsi-fungsi yang ada pada sistem tersebut telah berjalan dengan baik dan sesuai dengan kebutuhan pengguna.

2.1.9. Pembayaran Siswa

Istilah pembayaran berarti bahwa kegiatan pembayaran adalah sistem yang mencakup seperangkat aturan, lembaga dan mekanisme yang digunakan untuk melaksanakan pemindahan dana

guna memenuhi suatu kewajiban yang timbul dari suatu kegiatan ekonomi (Teguh, 2004). Pembayaran adalah suatu tindakan menukarkan sesuatu (uang/barang) dengan maksimal dan tujuan yang sama yang dilakukan oleh dua orang atau lebih (Waluyo, 2007). SPP merupakan iuran wajib bagi siswa-siswi yang dipergunakan oleh pihak sekolah untuk memfasilitasi segala kegiatan pembelajaran yang dilakukan siswa-siswi dengan waktu pembayaran ditentukan sebelumnya. SPP dapat diartikan sebagai sumbangan pembinaan pendidikan yang merupakan iuran wajib bagi siswa siswi selama menjalani kegiatan sekolah yang harus di bayarkan setiap bulan demi kelancaran kegiatan sekolah (Kamus Besar Bahasa Indonesia, 2008:205). PSB adalah pembayaran pendaftaran bagi siswa baru yang dipergunakan oleh pihak sekolah untuk kegiatan pendaftaran siswa baru. Dibayarkan oleh siswa saat mendaftar. Uang pangkal ialah uang yang dibayarkan oleh siswa-siswi bilamana telah terdaftar menjadi siswa baru. Dapat dibayar secara tunai maupun secara angsur. Uang UTS/ UAS dibayarkan sebelum penyelenggaraan Ulangan Tengah Semester atau Ulangan Semester, yang dipergunakan oleh pihak sekolah untuk penyelenggaraan kegiatan Ulangan Umum.

3. Metodologi

Tahap-tahap penelitian yang digunakan pada Pengembangan Aplikasi Pembayaran Siswa adalah: Studi Literatur, Wawancara dan observasi. Metode penelitian yang digunakan adalah Penelitian eksperimen. Berdasarkan pada studi pustaka, pengumpulan data dan analisis masalah. Penulis berkesimpulan untuk mengefisienkan waktu dalam perancangan dan desain serta untuk meminimalisir keadaan miss komunikasi antara *software developer* dan *customer* maka pada pengembangan perangkat lunak ini menggunakan metode *prototype* dalam SDLC.

Table 3.1 Daftar *Requirement* untuk Pengguna *Super Admin* Tahap 1

No.	Requirement	Use Case
<i>Actor : Super Admin</i>		
1	<i>Super Admin</i> dapat melakukan login sebagai <i>Super Admin</i>	Login
2	<i>Super Admin</i> dapat input kelas	Bagi kelas
3	<i>Super Admin</i> dapat melakukan input unit	Bagi unit
4	<i>Super Admin</i> dapat melakukan input kepala sekolah	Input Kepsek
5	<i>Super Admin</i> dapat melakukan input Wali Kelas	Input Walas
6	<i>Super Admin</i> dapat melakukan input data siswa	Input Data Siswa
7	<i>Super Admin</i> dapat melakukan aktifasi siswa	Aktifasi Siswa
8	<i>Super Admin</i> dapat melakukan input Kasir	Input Kasir
9	<i>Super Admin</i> dapat melakukan input Tagihan	Input Tagihan

Catatan: Superadmin didaftar secara langsung di *database*.

Table 3.2. Daftar *Requirement* untuk Pengguna Kasir Tahap 1

No.	Requirement	Use Case
<i>Actor : Kasir</i>		
1	Kasir dapat melakukan login sebagai Kasir	Login
2	Kasir dapat memasukan data uang pembayaran	Input Pembayaran
3	Kasir dapat mencetak bukti pembayaran	Cetak Bukti Pembayaran
4	Kasir dapat melihat tagihan siswa	Lihat Tagihan

Table 3.3 Daftar *Requirement* untuk Pengguna Yayasan Tahap 1

No.	Requirement	Use Case
<i>Actor : Yayasan</i>		
1	Yayasan dapat melakukan login sebagai Yayasan.	Login
2	Yayasan dapat melihat laporan pembayaran dari seluruh unit per bulan.	Lihat laporan

Table 3.4 Daftar *Requirement* untuk Pengguna Kepala Sekolah Tahap 1

No.	Requirement	Use Case
<i>Actor : Kepala Sekolah</i>		
1	Kepala Sekolah dapat melakukan login sebagai Kepala Sekolah.	Login
2	Kepala Sekolah dapat melihat laporan pembayaran dari seluruh kelas di unitnya per bulan.	Lihat laporan

Table 3.5 Daftar *Requirement* untuk Pengguna Wali Kelas Tahap 1

No.	Requirement	Use Case
<i>Actor : Wali Kelas</i>		
1	Wali Kelas dapat melakukan login sebagai Wali Kelas	Lihat laporan
2	Wali Kelas dapat melihat laporan pembayaran siswa di kelasnya per bulan	Lihat laporan

Table 3.6 Daftar *Requirement* untuk Pengguna *Super Admin* Tahap 2

No.	Requirement	Use Case
<i>Actor : Super Admin</i>		
1	<i>Super Admin</i> dapat melihat daftar siswa yang memiliki piutang	Melihat Piutang

Table 3.7 Daftar *Requirement* untuk Pengguna Yayasan Tahap 2

No.	Requirement	Use Case
<i>Actor : Yayasan</i>		
2	Yayasan dapat melihat laporan piutang siswa dari seluruh unit per bulan.	Melihat Piutang

Table 3.8 Daftar *Requirement* untuk Pengguna Kepala Sekolah Tahap 2

No.	Requirement	Use Case
<i>Actor : Yayasan</i>		
1	Kepala Sekolah dapat melihat laporan piutang siswa dari seluruh kelas di unitnya per bulan.	Melihat Piutang

Table 3.9 Daftar *Requirement* untuk Pengguna Wali Kelas Tahap 2

No.	Requirement	Use Case
<i>Actor : Wali Kelas</i>		
1	Wali Kelas dapat mencetak laporan pembayaran siswa di kelasnya per bulan.	Cetak Laporan Pembayaran
2	Wali Kelas dapat melihat laporan piutang siswa di kelasnya.	Lihat Piutang

Table 3.10 Daftar *Requirement* untuk Pengguna Siswa Tahap 2

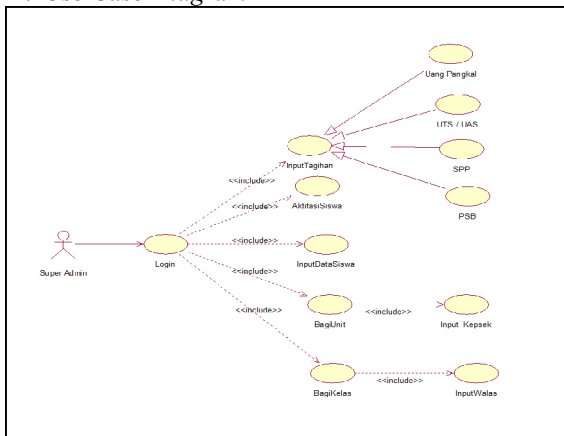
No.	Requirement	Use Case
Actor : Siswa		
1	Siswa dapat melakukan <i>Login</i> sebagai Siswa.	<i>Login</i>
2	Siswa dapat melihat tagihan pembayaran.	Lihat Tagihan

4. Hasil dan Analisis

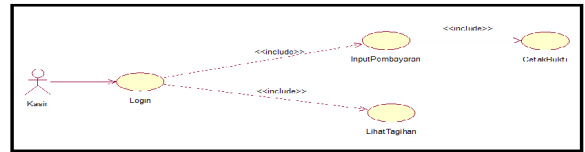
4.1. Hasil Analisis Kebutuhan

Berikut adalah beberapa analisis masalah yang harus dijawab oleh *software*: cara mengotomasi proses akses pembayaran siswa, Memberikan akses admin untuk memasukan, merubah, menghapus data siswa dan menyimpan hasil dari transaksi pembayaran siswa. Memberikan akses kepada siswa untuk melihat tagihan, cara *input* data siswa, cara melihat tagihan, cara *input* pembayaran, cara melihat laporan pembayaran, cara mencetak bukti pembayaran. Berikut adalah hasil dari pengujian *software* atas permasalahan tersebut: Masalah mengotomasi proses akses Aplikasi Pembayaran Siswa dijawab dengan adanya sistem *login* dan hak akses pada proses masuk. Siswa mengakses aplikasi sebagai *user* dan pihak Yayasan (*Super Admin*, Kasir, Wali Kelas, Kepala Sekolah, dan Yayasan) mengakses sebagai admin sesuai dengan hak aksesnya masing-masing, pemberian hak akses pada *Super Admin* untuk memasukan, merubah, dan menghapus setiap data bilamana diperlukan, cara kerja *input* data siswa adalah dengan cara memilih menu pendaftaran dan ikon ubah *biodata* untuk merubah data yang salah pada *form* yang sudah disediakan. Data yang sudah terinput pada *form* akan tersimpan di *database*, cara kerja melihat tagihan adalah dengan cara memilih menu tagihan, data yang tersimpan pada *database* akan ditampilkan oleh sistem, cara kerja *input* pembayaran adalah dengan cara memilih menu pembayaran, cara kerja melihat laporan pembayaran adalah dengan cara memilih menu Laporan. Bukti pembayaran dapat dicetak bila telah melakukan *input* pembayaran.

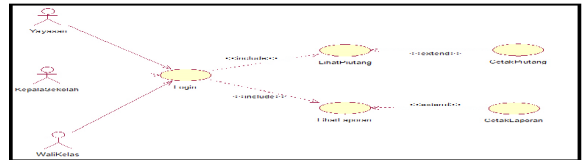
A. Use Case Diagram



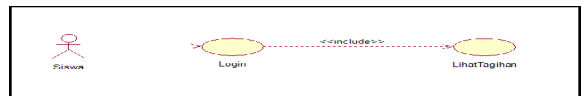
Gambar 4.1. Use Case Diagram Super Admin



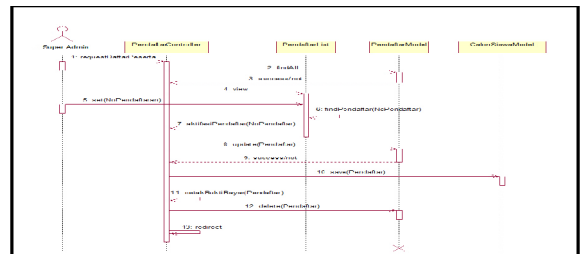
Gambar 4.2. Use Case Diagram Kasir



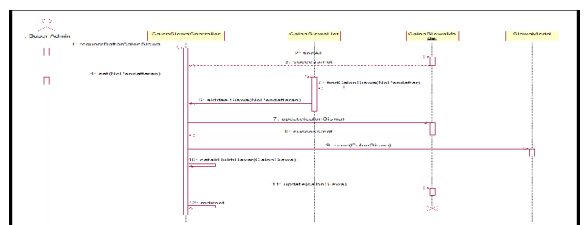
Gambar 4.3. Use Case Diagram Wali Kelas, Kepala Sekolah dan Yayasan



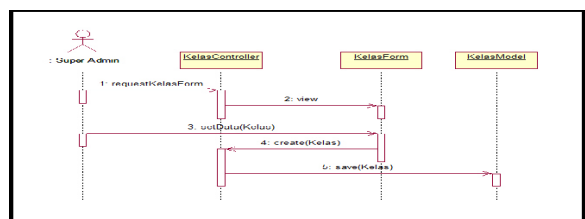
Gambar 4.4. Gambar Use Case Diagram Siswa



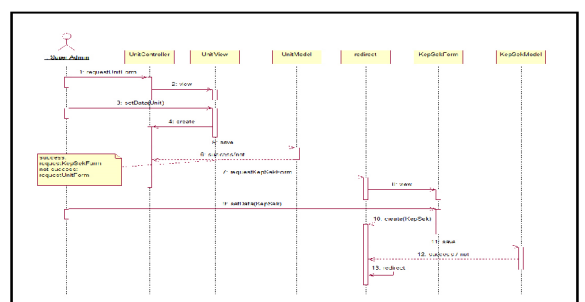
Gambar 4.5. Sequence Diagram Lihat Data Pendaftaran



Gambar 4.6. Sequence Aktivasi Siswa



Gambar 4.7. Sequence Input Kelas Baru



Gambar 4.8. Sequence Input Unit Baru

4.3 Struktur Data

Tabel 4.1. Jabatan

No.	Nama field	Tipe Data	Attributes	Nulls	Extra	Index
1	id_jabatan	char(1)		No		
2	jabatan	varchar(45)		Yes		

Tabel 4.2. Jenis Pembayaran

No.	Nama field	Tipe Data	Attributes	Nulls	Extra	Index
1	id_jns_pembayaran	char(1)		No		
2	jns_pembayaran	varchar(45)		Yes		
3	jns_pembayaran_ab	varchar(3)		No		

Tabel 4.3. Kelas

No.	Nama field	Tipe Data	Attributes	Nulls	Extra	Index
1	id_kelas	int(10)	UNSIGNED	No	auto_increment	primary_key
2	kelas	varchar(45)		Yes		
3	unit	tinyint(3)	UNSIGNED	No		foreign_key
4	wali_kelas	int(10)	UNSIGNED	No		foreign_key
5	tingkat	Char(1)		No		

Tabel 4.4. Pegawai

No.	Nama field	Tipe Data	Attributes	Nulls	Extra	Index
1	nama_pegawai	varchar(45)		No		
2	nuptk	varchar(16)		Yes		
3	nik	Char(18)		No		
4	tgl_lahir	date		No		
5	jenis_kel	char(1)		No		
6	alamat	varchar(75)		No		
7	id_username	int(10)	UNSIGNED	No		foreign_key
8	id_jabatan	Char(1)		No		foreign_key
9	nip	Char(18)		No		
10	photo	Char(36)		No		

Tabel 4.5. Pembayaran

No.	Nama field	Tipe Data	Attributes	Nulls	Extra	Index
1	id_pembayaran	int(10)	UNSIGNED	No	auto_increment	primary_key
2	tgl_pembayaran	datetime		Yes		
3	siswa	int(10)	UNSIGNED	No		foreign_key
4	kasir	int(10)	UNSIGNED	No		foreign_key
5	jenis_pembayaran	char(1)		No		foreign_key
6	jml_bayaran	float		Yes		
7	potongan	float		Yes		
8	nomorKwitansi	Char(12)		No		
9	tgl_tutup_pembayaran	date		Yes		
10	jml_dibayar	float		Yes		
11	angsur_no	int(11)		No		

Tabel 4.6. Role

No.	Nama field	Tipe Data	Attributes	Nulls	Extra	Index
1	id_role	char(1)		No		
2	role	varchar(45)		Yes		

Tabel 4.7. Piutang

No.	Nama field	Tipe Data	Attributes	Nulls	Extra	Index
1	id_piutang	int(11)	UNSIGNED	No	auto_increment	primary_key
2	tgl_piutang	varchar(45)		Yes		
3	siswa	int(10)	UNSIGNED	No		foreign_key
4	kasir	int(10)	UNSIGNED	No		foreign_key
5	jenis_pembayaran	char(1)		No		foreign_key
6	jml_piutang	float		Yes		

Tabel 4.8. Piutang Set Pembayaran

No.	Nama field	Tipe Data	Attributes	Nulls	Extra	Index
1	id_set_pembayaran	int(11)		No	auto_increment	primary_key
2	jum_pembayaran	float		Yes		
3	tgl_akhir_pembayaran	date		Yes		
4	tenis_pembayaran	char(1)		No		foreign_key
5	unit	tinyint(3)	UNSIGNED	No		foreign_key
6	aktif	char(1)		Yes		

Tabel 4.9. Unit

No.	Nama field	Tipe Data	Attributes	Nulls	Extra	Index
1	id_unit	tinyint(3)	UNSIGNED	No	auto_increment	primary_key
2	unit	varchar(45)		Yes		
3	kepala_sekolah	int(10)	UNSIGNED	No		foreign_key

Tabel 4.10. Username

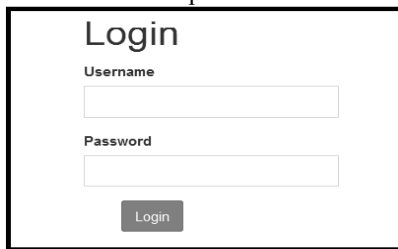
No.	Nama field	Tipe Data	Attributes	Nulls	Extra	Index
1	id_username	int(10)	UNSIGNED	No	auto_increment	primary_key
2	username	varchar(45)		Yes		
3	password	char(40)		Yes		
4	username_role	char(1)		No		foreign_key

Tabel 4.11. Siswa

No.	Nama field	Tipe Data	Attributes	Nulls	Extra	Index
1	nama	varchar(45)		Yes		
2	jns_kelamin	char(1)		Yes		
3	agama	Char(1)		Yes		
4	tmp_lahir	varchar(45)		Yes		
5	tgl_lahir	varchar(45)		Yes		
6	alamat	varchar(45)		Yes		
7	no_telp	varchar(20)		Yes		
8	gol_darah	varchar(2)		Yes		
9	anak_ka	varchar(2)		Yes		
10	ketwarganegaraan	varchar(2)		Yes		
11	nik	char(16)		Yes		
12	ms	char(8)		No		
13	nisan	char(10)		Yes		
14	no_uasbn	char(14)				
14	no_attb	char(10)		Yes		
15	nilai_skhun	Float		Yes		
16	sekolah_asal	varchar(45)		Yes		
17	tinggi_bdn	Float		Yes		
18	berat_badan	Float		Yes		
19	pnlykt	varchar(45)		Yes		
20	nama_ayah	varchar(25)		Yes		
21	pkrjaan_ayah	char(1)		Yes		
22	pendidikan_ibu	varchar(2)		Yes		
23	penghasilan_ortu	char(1)		Yes		
24	alamat_ortu	varchar(45)		Yes		
25	nama_wali	varchar(25)		Yes		
26	pkrjaan_wali	char(1)		Yes		
27	pendidikan_wali	varchar(2)		Yes		
28	penghasilan_wali	char(1)		Yes		
29	alamat_wali	varchar(45)		Yes		
30	kelas	int(10)	UNSIGNED	No		foreign_key
31	id_username	int(10)	UNSIGNED	No		foreign_key
32	tahun_angkatan	char(4)		No		
33	potongan_spp	float		Yes		
34	potongan_uas	float		Yes		
35	potongan_uts	float		Yes		
36	potongan_up	float		Yes		
37	photo	char(36)		No		
38	yatim_piatu	char(1)		Yes		
39	skhun	char(9)		Yes		
40	tempat_lahir_ayah	varchar(150)		No		
41	tempat_lahir_ibu	varchar(150)		No		
42	tempat_lahir_wali	varchar(150)		No		
43	tanggal_lahir_ayah	date		No		
44	tanggal_lahir_ibu	date		No		
45	tanggal_lahir_wali	date		No		
46	jum_saudara	varchar(2)		Yes		
47	tahun_attb	char(4)		Yes		
48	almt_sekolah_asal	varchar(150)		Yes		

4.4 Implementasi Hasil Tampilan

Halaman utama aplikasi



Gambar 4.18. Halaman Utama Aplikasi

Halaman utama aplikasi adalah halaman *Login*. Untuk masuk ke aplikasi pembayaran sekolah, *user* harus melakukan *login* sesuai dengan hak aksesnya yang telah terdaftar pada aplikasi pembayaran siswa. Berikut ini tampilan dari halaman utama.



Gambar 4.19. Halaman Beranda Super Admin

Pada halaman *Superadmin* memiliki beberapa fungsi utama diantaranya untuk pendaftaran akun bagi pegawai sebagai admin sesuai dengan hak aksesnya yaitu: admin kasir, admin wali kelas, admin kepala sekolah dan admin yayasan, pendaftaran akun bagi siswa, akun tersebut digunakan untuk login di aktivitas selanjutnya sesuai dengan hak aksesnya. *Input Angkatan*, *Input Unit*, *Input Kelas*, membuat aturan pembayaran dan menghapus data atau *update* data.



Gambar 4.20. Halaman Beranda Kasir

Pada halaman beranda kasir berbeda dengan halaman beranda *superadmin* pada halaman beranda kasir yang tampil hanya menu pembayaran dan tagihan memiliki fungsi untuk melakukan transaksi pembayaran siswa



Gambar 4.21. Halaman Beranda Wali Kelas

Halaman Beranda Wali Kelas memiliki fungsi untuk mengetahui laporan pembayaran dan tagihan siswa di kelas.



Gambar 4.22. Halaman Beranda Kepala Sekolah

Pada halaman Beranda Kepala Sekolah memiliki fungsi untuk mengetahui laporan pembayaran dan tagihan siswa di unit



Gambar 4.23. Halaman Beranda Yayasan

Halaman Beranda Yayasan memiliki fungsi untuk mengetahui laporan pembayaran dan tagihan siswa di semua unit yang ada di Yayasan Perguruan Islam As Sa'adah.

4.5 Pengujian

Berdasarkan hasil pengujian UAT yang telah dilakukan kepada 3 *user* kasir, 8 *user* siswa, 4 *user* kepek, 4 *user* wali kelas, 1 *user* yayasan dan 1 *user* superadmin, maka diperoleh hasil pengujian aplikasi pembayaran siswa pada Yayasan Perguruan Islam As Sa'adah tersebut dengan hasil kesimpulan diterima dari setiap pengujian yang telah dilakukan dan telah sesuai dengan kriteria atau analisis kebutuhan pengguna dan bebas dari kesalahan. Hasil pengujian Aplikasi Pembayaran Siswa tersebut telah di periksa dan disetujui oleh Ketua Yayasan Perguruan Islam As Sa'adah, sebagaimana yang tertera pada lembar pengesahan pengujian

5. Kesimpulan dan Saran

5.1. Kesimpulan

Berdasarkan pembahasan dan analisa pada bab sebelumnya maka dalam Pengembangan Aplikasi Pembayaran Siswa dengan menggunakan metode prototype dan setelah Pengujian kelayakan aplikasi diuji oleh *user* dengan menggunakan metode *User Acceptance Test* (UAT). Berdasarkan hasil pengujian UAT yang telah dilakukan kepada 3 *user* kasir, 8 *user* siswa, 4 *user* kepek, 4 *user* wali kelas, 1 *user* yayasan dan 1 *user* superadmin, maka diperoleh hasil pengujian aplikasi pembayaran siswa pada Yayasan Perguruan Islam As Sa'adah tersebut dengan hasil kesimpulan diterima dari setiap pengujian yang telah dilakukan dan telah sesuai dengan kriteria atau analisis kebutuhan pengguna dan bebas dari kesalahan. Hasil pengujian Aplikasi Pembayaran Siswa tersebut telah di periksa dan disetujui oleh Ketua Yayasan Perguruan Islam As Sa'adah Pondok Kelapa, sebagaimana yang tertera pada lembar pengesahan pengujian. Dengan demikian aplikasi pembayaran siswa tersebut dapat dimanfaatkan oleh Yayasan Perguruan Islam As Sa'adah Pondok Kelapa.

5.2. Saran

Beberapa saran untuk menjadi masukan agar aplikasi dapat digunakan dengan optimal, yaitu: Sebaiknya dalam setiap pergantian tahun ajaran dilakukan evaluasi terhadap kinerja Aplikasi Pembayaran Siswa. Evaluasi tersebut mencakup penilaian kebutuhan dan performa sistem. Apabila kompleksitas informasi yang dibutuhkan meningkat, dapat dilakukan pengembangan lanjutan terhadap Aplikasi tersebut. Melakukan *Backup* data secara berkala terhadap *database* untuk menjaga kestabilan aplikasi dan mengurangi resiko kehilangan data akibat gangguan teknis dengan cara : a) Menyimpan data pada perangkat lain, seperti *flashdisk*, *external hard disk*, *CD*. b) Menyimpan data pada *e-mail*, *Dropbox*, *Google Drive*.

Daftar Pustaka:

- Bentley, Lonnie D, dan Jeffrey L Whitten. 2007. *Systems Analysis and Design for the Global Enterprise Seventh Edition*, New York: McGraw-Hill.
- Black, Rex. 2002. *Managing the Testing Process: Practical Tolls and Techniques for Managing Hardware and Software Testing, 2nd Edition*. Hoboken: Wiley Publishing Inc.
- Black, Rex. 2007. *Pragmatic Software Testing: Becoming an Effective and Efficient Test Profesional*, Hoboken: Wiley Publishing Inc.
- Brooks, F. 1975. *The Mythical Man-Moth*, Addison-Wesley

- Departemen Pendidikan Nasional. 2008. *Kamus Bahasa Indonesia*. Jakarta: Pusat Bahasa.
- [FT] Fakultas Teknik. 2012. *Buku Pedoman Skripsi/ Komprehensif/ Karya Inovatif (SI)*. Jakarta: Fakultas Teknik, Universitas Negeri Jakarta.
- Harip Santoso. 2000. *Pengertian Aplikasi*. <http://dilihatya.com/1178/pengertian-aplikasi-menurut-para-ahli>. [17 Juli 2015]
- Hendrayudi . 2004. Arti Aplikasi.[terhubung berkala]. <http://dilihatya.com/1178/pengertian-aplikasi-menurut-para-ahli>. [17 Juli 2015]
- Ibisa. 1999. Pengertian Aplikasi. [terhubung berkala] <http://dilihatya.com/1178/pengertian-aplikasi-menurut-para-ahli>. [17 Juli 2015]
- [IEE93] IEEE. 1993. *Standards Collection: Software Engineering, IEEE Standard 610.12-1990*.
- Jogiyanto. 1999. *Definisi Aplikasi*. [terhubung berkala]. <http://dilihatya.com/1178/pengertian-aplikasi-menurut-para-ahli>. [17 Juli 2015]
- Jovanovic, Irena. 2009. *Software Testing Methods and Techniques. Journal of Computer Science* . 5(6). Hlm. 30.
- Lewis, W.E. 2009. *Software Testing and Continuous Quality Improvement*, Third Edition. Boston: Auerbach Publication.
- Marlinda, Linda. 2004. *Sistem basis data*. Yogyakarta: Andi Offset.
- Naur, P., dan B, Randall (eds). 1969. *Rekayasa Perangkat Lunak: A Report on a Conference Sponsored by NATO Science Committee*. NATO.
- Pakpahan, Nasib. 2015. *Pengertian Dan Definisi Aplikasi Menurut Para Ahli*. [Terhubung berkala]. <http://blogdefinisi.blogspot.co.id/2015/08/pengertian-dan-definisi-aplikasi.html>. [30 Agustus 2015].
- Perry, William E. 2006. *Effective Methods for Software Testing, 3rd Edition*. Wiley Publishing, Inc. India Napolis, Indiana.
- Pressman, R.S. 2010. *Software Engineering : a practitioner's approach*. New York: McGraw-Hill.
- Santoso, Harip. 2000. *Pengertian Aplikasi*. [terhubung berkala]. <http://dilihatya.com/1178/pengertian-aplikasi-menurut-para-ahli>. [17 Juli 2015]
- Sommerville, Ian. 2011. *Software Engineering, 9th Edition*. Boston: Addison-Wesley.
- Swastika Windra, 2006. *PHP 5 dan MySQL 4, Proyek Membuat Blog*. Jakarta.PT. Dian Rakyat.
- Teguh, W. 2004. *Sistem Basis Data*. Bandung: graha ilmu.
- Waluyo. 2007. *Perpajakan Indonesia*, Jakarta: Salemba Empat , Edisi Kedua.