

BAB II

TINJAUAN PUSTAKA

2.1 Sistem Informasi Administrasi

Menurut Agus and Indera (2017) Sistem informasi merupakan hasil dari suatu tahapan perencanaan yang menjadi pedoman dalam pembangunan atau pengembangan sistem informasi seperti melakukan perencanaan strategis sistem informasi untuk mendapatkan masukan tentang kebutuhan sistem informasi yang sesuai dengan lingkungan organisasi atau proses bisnis organisasi, dan dapat mengalokasikan sumber daya yang dimiliki oleh organisasi.

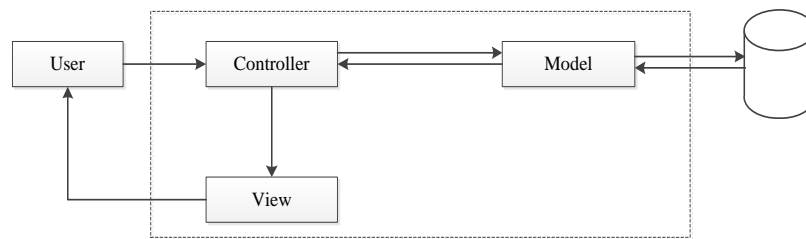
Menurut Anisah and Novianti (2017) Administrasi adalah sebuah sistem dimana didalamnya terdapat perencanaan, pengendalian, dan pengorganisasian. Selain itu administrasi sebagai *rule of game* dari suatu pekerjaan baik perkantoran, agar target target dalam perencanaan bisa tercapai secara maksimal dan berfungsi dari bagaimana cara melakukan komunikasi dalam bentuk pelayanan suatu kumpulan, organisasi atau struktur perkumpulan itu sendiri

Menuru Sumiati *et al.*, (2020) Administrasi secara luas diartikan sebagai data dan informasi yang di susun ataupun dicatat secara terstruktur baik eksternal ataupun internal. Tujuan dari pengumpulan data tersebut dengan maksud agar untuk mendapatkan data tersebut mudah untuk di olah baik sebagian ataupun secara keseluruhan. sedangkan secara sederhana administrasi adalah tata usaha.

2.2 CodeIgniter

Menurut Raharjo (2018), *CodeIgniter* adalah *Framework* untuk bahasa pemrograman PHP, yang dibuat Rick Ellis pada tahun 2006. *CodeIgniter* memiliki banyak fitur yang membantu para pengembang PHP untuk dapat membuat aplikasi secara mudah dan cepat serta memiliki sifat yang fleksibel dapat mengembangkan dalam perangkat *web*, dekstop maupun *mobile*".

CodeIgniter memiliki konsep atau pola *Model-View-Controller* (MVC) sehingga kode-kode dapat di sederhanakan.



Gambar 2.1 Arsitektur MVC

2.2.1 Web Based

Menurut Urbietta *et al*, (2019), *Web Based* adalah aplikasi yang dibuat berbasis *web* yang membutuhkan *web server* dan *browser* untuk menjalankannya.

Menurut Purwati *et al*, (2018), Dengan membuat sistem berbasis *web based* ada beberapa hal yang penting dan harus kita pikirkan sebelum membangun sistem tersebut, diantaranya:

1. Tidak membutuhkan *hardware* dengan spesifikasi yang tangguh untuk menjalankan aplikasinya.
2. Server yang dibutuhkan cukup diinstallkan *tools* pendukung saja agar klien mudah menjalankan aplikasi
3. Infrastruktur jaringan yang dibutuhkan juga cukup besar karena aplikasi yang dibuat dapat diakses dari jaringan luar (internet).
4. Aplikasi berbasis *web based* dapat diakses dari berbagai perangkat dengan syarat menggunakan *web browser* saja sudah dapat mengaksesnya.
5. Jika aplikasi yang sudah jadi ingin di *update*, sangat mudah untuk melakukannya karena tidak membutuhkan membuka keseluruhan aplikasi.

2.2.2 PHP

Menurut Subagja (2018), PHP adalah bahasa *server-side-scripting* yang menyatudengan HTML untuk membuat halaman *web* yang dinamis.

Menurut Aryani *et al.*, (2015), berpendapat bahwa *PHP Hypertext Preprocessor* adalah bahasa pemrograman *web server-side* yang bersifat *open source*. PHP merupakan *script* yang terintegrasi dengan HTML dan berada pada *server* (*server side HTML embedded scripting*). PHP adalah *script* yang digunakan

untuk membuat halaman *website* yang dinamis. Dinamis berarti halaman yang akan ditampilkan dibuat saat halaman itu diminta oleh *client*. Mekanisme ini menyebabkan informasi yang diterima *client* selalu yang terbaru/*up to date*. Semua script PHP dieksekusi pada *server* dimana script tersebut dijalankan. Dengan menggunakan program PHP, sebuah *website* akan lebih interaktif dan dinamis.

Sehingga PHP merupakan bahasa pemrograman yang digunakan oleh pengembang untuk membuat sistem *website* dengan kumpulan bahasa HTML dan *script* lainnya.

2.2.3 MySql

Menurut MySQL (2018), *MySQL* adalah singkatan dari *Structue Query Language* yang digunakan untuk mendefinisikan structure data, memodifikasi data pada basis data, menspesifikasi batasan keamanan (*security*), hingga pemeliharaan data.

Menurut Amin (2018), mendefinisikan *mysql* adalah RDBMS yang cepat dan mudah digunakan, serta sudah banyak digunakan untuk berbagai kebutuhan.

MySQL merupakan bahasa standar yang paling banyak digunakan untuk mengakses *database* relasional dan merupakan aplikasi yang dapat dipergunakan secara bebas.

2.3 Metode Pengembang Sistem

Metode pengembang sistem merupakan metode yang digunakan sebagai alur proses dalam pengembangan, sehingga penelitian dapat di kembangkan sesuai tahapan dari metode pengembang sistem.

2.3.1. *Rapid Application Development (RAD)*

Menurut Rosa and Shalahuddin (2019), RAD merupakan model proses pengembangan perangkat lunak yang bersifat incremental terutama untuk waktu pengerjaan yang pendek. Model RAD adalah adaptasi dari model air terjun versi kecepatan tinggi, dengan menggunakan model air terjun untuk pengembangan setiap komponen perangkat lunak. Keuntungan utama menjalankan *rapid application development* adalah jangka waktu pengembangan lebih cepat.



Gambar 2.2 *Rapid Application Development (RAD)*
 Sumber: (Rosa and Shalahuddin, 2019)

1. Kelebihan Model RAD

Kelebihan metodologi RAD:

- Penghematan waktu dalam keseluruhan fase proyek dapat dicapai.
- RAD mengurangi seluruh kebutuhan yang berkaitan dengan biaya proyek dan sumberdaya manusia.
- RAD sangat membantu pengembangan aplikasi yang berfokus pada waktu penyelesaian proyek.
- Perubahan desain sistem dapat lebih berpengaruh dengan cepat dibandingkan dengan pendekatan SDLC tradisional.
- Sudut pandang user disajikan dalam sistem akhir baik melalui fungsi-fungsi sistem atau antarmuka pengguna.
- RAD menciptakan rasa kepemilikan yang kuat di antara seluruh pemangku kebijakan proyek.

2. Kelemahan Model RAD

Kelemahan pada pengembangan tersebut dapat dilihat berdasarkan kesesuaian pengembangan yang dilakukan, berikut adalah kelemahan metode pengembang sistem RAD :

- Dengan metode RAD, penganalisis berusaha mempercepat proyek dengan terburu-buru.

- b. Kelemahan yang berkaitan dengan waktu dan perhatian terhadap detail. Aplikasi dapat diselesaikan secara lebih cepat, tetapi tidak mampu mengarahkan penekanan terhadap permasalahan-permasalahan perusahaan yang seharusnya diarahkan.
- c. RAD menyulitkan *programmer* yang tidak berpengalaman menggunakan prangkat ini di mana *programmer* dan *analyst* dituntut untuk menguasai kemampuan-kemampuan baru sementara pada saat yang sama mereka harus bekerja mengembangkan sistem

2.3.2 Tahapan Penelitian

Tahapan dalam penelitian sebagai langkah-langkah penelitian yang harus dikerjakan, berikut adalah tahapan penelitian Rapid Application Development.

1. Tahap Pemodelan Bisnis

Pemodelan yang dilakukan untuk memodelkan fungsi bisnis untuk mengetahui informasi apa yang terkait proses bisnis, informasi apa saja yang harus dibuat, siapa yang membuat informasi, bagaimana alur informasinya, proses apa yang terkait informasi tersebut.

2. Tahap Pemodelan Data

Memodelkan data apa saja yang dibutuhkan berdasarkan pemodelan bisnis dan mendefinisikan atribut-atributnya beserta relasinya dengan data-data yang lain.

3. Tahap Pemodelan Proses

Mengimplementasikan fungsi bisnis yang sudah didefinisikan terkait dengan pendefinisian data.

4. Tahap Pembuatan Aplikasi

Mengimplementasikan pemodelan proses dan data menjadi program. Model RAD sangat menganjurkan pemakaian komponen yang sudah ada jika dimungkinkan.

5. Tahap Pengujian dan Pergantian

Menguji komponen-komponen yang dibuat. Jika sudah teruji maka tim pengembang komponen dapat beranjak untuk mengembangkan komponen berikutnya.

2.4 Alat Pengembang Sistem

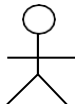

Menurut Rosa dan Salahuddin (2019), *Unified Modelling Language* adalah bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teks-teks pendukung. Berikut ini merupakan penjelasan tentang masing-masing diagram yang ada pada *Unified Modelling Language*.

Menurut Kurniawan, Karnila and Indera (2021), *Object Oriented Programming* (OOP) adalah suatu strategi pembangunan perangkat lunak yang mengorganisasikan perangkat lunak sebagai kumpulan objek yang berisi data dan operasi yang diberlakukan terhadapnya



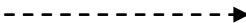
2.4.1 Use Case Diagram

Menurut Rosa dan Salahuddin (2019), *use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. *Use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Berikut simbol-simbol yang akan digunakan dalam menggambarkan *Use Case Diagram* dapat dilihat pada Tabel 2.1 :

Tabel 2.1 *Simbol-simbol Use Case Diagram*

No	Simbol	Keterangan Fungsi
1.	Aktor 	Aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu.
2.	<i>Use Case</i> 	<i>Use Case</i> adalah deskripsi dari urutan aksi- aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor.

Tabel 2.1 *Simbol-simbol Use Case Diagram (Lanjutan)*



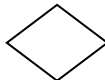

No	Simbol	Keterangan Fungsi
3.	Asosiasi 	Asosiasi adalah apa yang menghubungkan antara objek satu dengan objek yang lainnya.
4.	Generalisasi 	Generalisasi adalah hubungan dimana objek anak (<i>descendent</i>) berbagi perilaku dan struktur data dari objek yang ada di atasnya atau sebaliknya dari bawah ke atas.
5.	Defendancy 	<i>Defendancy</i> (ketergantungan) adalah hubungan dimana perubahan yang terjadi pada suatu elemen defenden (mandiri) akan mempengaruhi elemen yang bergantung padanya (independen).

Sumber: (Rosa dan Salahuddin, 2019)

2.4.2 Activity Diagram

Menurut Rosa dan Salahuddin (2019), *activity diagram* menggambarkan aliran kerja atau aktivitas dari sebuah sistem atau proses bisnis atau menggambarkan aktivitas sistem bukan apa yang dilakukan aktor. Berikut simbol-simbol yang akan digunakan dalam menggambarkan *activity diagram* dapat dilihat pada Tabel 2.2:

Tabel 2.2 Simbol Activity Diagram

No.	Simbol	Keterangan
1.		Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal.
2.		Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja.
3.		Percabangan (<i>Decision</i>) merupakan asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu.
4.		Penggabungan (<i>Join</i>) merupakan asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu.

Tabel 2.2 Simbol Activity Diagram (Lanjutan)

No.	Simbol	Keterangan
-----	--------	------------

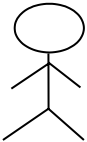

5.	Nama swimlane	Swimlane Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas.

Sumber: (Rosa dan Salahuddin, 2019)

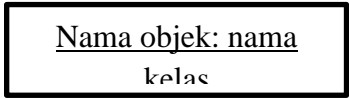

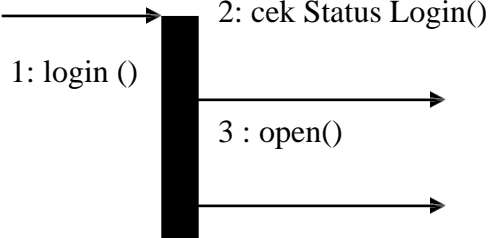
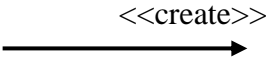

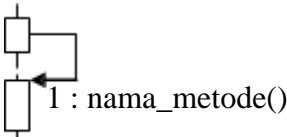

2.4.3 *Sequence Diagram*

Diagram *sequence* menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dengan *message* yang dikirimkan dan diterima antar objek. Oleh karena itu untuk menggambarkan diagram sekuen maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu. Membuat diagram sekuen juga dibutuhkan untuk melihat skenario yang ada pada *use case*. Banyaknya diagram sekuen yang harus digambar adalah minimal sebanyak pendefinisian *use case* yang memiliki proses sendiri atau yang penting semua *use case* yang telah didefinisikan interaksinya pesan sudah dicakup dalam diagram sekuen sehingga semakin banyak *use case* yang didefinisikan maka diagram sekuen yang harus dibuat juga semakin banyak. Berikut adalah simbol-simbol yang ada pada *sequence diagram* pada Tabel 2.3.

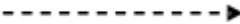
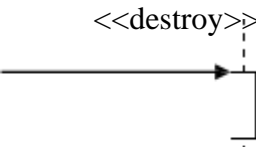
Tabel 2.3 Simbol-simbol *Sequence Diagram*

No.	Simbol	Deskripsi
1	<p>Aktor</p>  <p>Atau</p> <p><u>Nama</u></p> <p>Tanpa waktu aktif</p>	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang, biasanya dinyatakan dalam menggunakan kata benda di awal frase nama aktor
2.	<p>Garis hidup /<i>lifeline</i></p> 	Menyatakan kehidupan suatu objek

Tabel 2.3 Simbol-simbol *Sequence Diagram* (Lanjutan)

No.	Simbol	Deskripsi
3.	Objek 	Menyetakan objek yang berinteraksi peran
4.	Waktu aktif 	Menyatakan objek dalam keadaan aktif dan berinteraksi, semuanya yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan di dalamnya, misalnya  Maka cek status login() dan open() dilakukan didalam metode login(). Akor tidak memiliki waktu aktif.
5.	Pesan tipe <i>create</i> 	Menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat
6.	Pesan tipe <i>call</i> 1 : nama_metode() 	Menyatakan suatu objek memanggil oprasi/metode yang ada pada objek lain atau dirinya sendiri,  Arah panah mengarah pada objek yang memiliki operasi/metode, karena ini memanggil operasi/ metode, maka operasi/metode yang dipanggil harus ada pada diagram kelas sesuai dengan kelas objek yang berinteraksi.
7.	Pesan tipe <i>send</i> 1 : masukan 	Meyatakan bahwa suatu objek mengirimkan data/masukan/informasi/ ke objek lainnnnya, arah panah mengarah pada objek yang menerima kembalian.

Tabel 2.3 Simbol-simbol *Sequence Diagram* (Lanjutan)




No.	Simbol	Deskripsi
8.	Pesan tipe <i>return</i> 1: keluaran 	Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian.
9	Pesan tipe <i>destroy</i> 	Menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri, sebaliknya jika ada <i>create</i> maka ada <i>Destroy</i>

Sumber: (Rosa dan Salahuddin, 2019)

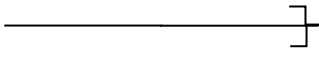
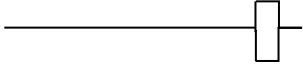
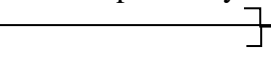
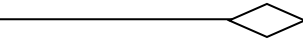
2.4.4 Class Diagram

Menurut Rosa dan Salahuddin (2019), *Class diagram* mengembangkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Berikut simbol-simbol yang akan digunakan dalam menggambarkan *Class Diagram* dapat dilihat pada Tabel 2.4:

Tabel 2.4 Simbol *Class Diagram*

No.	Simbol	Deskripsi
1.		Kelas pada struktur sistem.
2.	Antar Muka/Interface  Nama_Interface	Sama dengan konsep interface dalam pemrograman berorientasi objek.
3.	Asosiasi / Asociation 	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan simbol

Tabel 2.4 Simbol *Class Diagram* (Lanjutan)

No.	Simbol	Deskripsi
4.	Asosiasi Berarah / Directed Association 	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan symbol.
5.	Generalisasi 	Relasi antar kelas dengan makna generalisasi-spesialisasi (umum khusus)
6.	Ketergantungan / dependency 	Relasi antar kelas dengan makna ketergantungan antar kelas.
7.	Agregasi / aggregation 	Relasi antar kelas dengan makna semua bagian (<i>whole-part</i>)




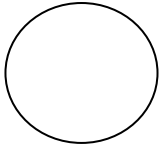
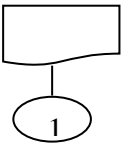
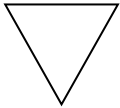
Sumber: (Rosa dan Salahuddin, 2019)

2.5 BAD (*Document Flowchart*)

Dengan menggunakan *flowchart*, langkah prosedur penyelesaian permasalahan dapat diekspresikan dengan serangkaian simbol grafis yang baku dan lebih mudah digunakan. Manfaat yang diperoleh menggunakan *flowchart* yaitu membiasakan berfikir secara sistematis dan terstruktur dalam setiap kesempatan serta lebih mudah mengecek dan menemukan bagian-bagian prosedur yang tidak valid.

Menurut (Nimas, 2016) *flowchart* adalah bagan (*chart*) yang menunjukkan alir (*flow*) di dalam program atau prosedur sistem secara logika. Simbol – simbol Bagan Alir Dokumen seperti berikut :

Tabel 2.5 Simbol – simbol Bagan Alir Dokumen

Simbol	Keterangan
 Dokumen	Simbol ini menggambarkan semua jenis dokumen yang merupakan formulir yang digunakan untuk merekam data terjadinya suatu transaksi.
 Dokumen dan Tembusannya	Simbol ini digunakan untuk menggambarkan dokumen asli dan tembusannya. Nomor lembar dokumen dicantumkan di sudut kanan atas.
 Catatan	Simbol ini digunakan untuk menggambarkan catatan akuntansi yang digunakan untuk mencatat data yang direkam sebelumnya dalam dokumen atau formulir.
 Connector	Simbol ini sebagai tanda penghubung bagan alir dokumen pada halaman yang sama, dengan memperhatikan nomor yang tercantum dalam simbol penghubung pada halaman yang sama.
 Awal arus dokumen	Berasal dari simbol penghubung halaman yang sama, yang bernomor seperti yang tercantum di dalam simbol tersebut.
 Arsip Sementara	Simbol ini digunakan untuk menunjukkan tempat penyimpanan dokumen seperti lemari arsip dan kotak arsip. Keterangan: A = Menurut abjad N = Menurut nomor urut T = Menurut tanggal / kronologis

2.6 Metode Pengujian Sistem

Metode pengujian sistem merupakan metode yang digunakan untuk melakukan testing pada sistem yang dibangun sehingga di peroleh hasil berupa sistem yang sesuai fungsinya.

2.6.1 Black Box

Black Box Testing merupakan pengujian yang dapat dilakukan dengan melakukan pengamatan, pada hasil eksekusi melalui beberapa data uji dan memeriksa fungsional yang terdapat pada perangkat lunak. Jadi dapat kita dianalogikan seperti halnya kita melihat ke dalam kotak hitam, sehingga kita hanya bisa melihat tampilan luarnya saja tanpa kita tau apa yang ada didalam kotak hitam tersebut.

Sehingga sama seperti halnya dengan *Black Box Testing* yang hanya dapat mengevaluasi dari tampilan luarnya dan fungsionalitasnya. Tanpa harus mengetahui apa sesungguhnya yang terjadi dalam proses detilnya. Pada pengetahuan khusus dari struktur kode internal dan pengetahuan pada pemrograman dasar pada umumnya tidak diperlukan untuk *Black Box Testing*. Uji pada kasus yang dibangun disekitar spesifikasi dan persyaratan, yakni pada aplikasi yang seharusnya dilakukan.

2.6.2 Tahapan Pengujian Sistem

Tahapan pengujian sistem digunakan untuk mengetahui proses pengujian yang akan dilakukan, berikut adalah tahapan pengujian *Black Box*:

1. *Descision Table*

Decision Table merupakan cara yang tepat untuk memodelkan logika yang cukup rumit, seperti diagram alur, *if-then-else* dan *switch* laporan kasus. Dalam kondisi ini mengaitkan dengan tindakan untuk melakukan, akan tetapi banyak kasus melakukannya dengan cara yang lebih elegan.

2. *All-Pairs Testing*

All-Pairs Testing atau disebut *pairwise testing* merupakan metode pengujian perangkat lunak kombinatorial yang digunakan untuk setiap pasangan parameter yang masuk kedalam sistem atau algoritma yang ada pada perangkat lunak.

3. *State Transition Table*

State Transition Table merupakan teori automata dan logika skuensial, pada table yang menunjukkan *state* dalam pengujian. Pada dasarnya sebuah table *state* merupakan table kebenaran yang digunakan untuk

beberapa *input* dan *output* termasuk dengan *state* berikutnya dengan kondisi yang sebenarnya terjadi.

4. *Equivalence Partitioning*

Equivalence Partitioning merupakan teknik yang membagi data masukan dari beberapa unit perangkat lunak menjadi beberapa partisi data dari mana *test case* dapat diturunkan. Pada prinsipnya, uji kasus ini dirancang untuk menutupi setiap partisi minimal.

Teknik ini digunakan untuk mendefinisikan kasus pengujian yang mengungkap kelas kesalahan, sehingga mengurangi jumlah pengujian yang harus dilakukan.

5. *Boundry Values Analysis*

Boundary Value Analysis merupakan Pengujian yang dirancang untuk mencakup perwakilan dari batas Nilai-nilai batas. Pada nilai-nilai di sebuah partisi kesetaraan atau sebesar nilai terkecil di kedua sisi tepi

2.7 Tinjauan Pustaka

Tinjauan Pustaka merupakan sumber literatur pada penelitian berdasarkan penelitian terdahulu, berikut data tinjauan Pustaka pada Tabel 2.3.

Tabel 2.6 Tinjauan Pustaka

No.	Nama dan tahun	Judul	Metode	Hasil
1	Arif Tri Atmaja, Djoko Santoso dan Patni Ninghardjanti (2018)	Penerapan Sistem Otomatisasi Admnistrasi Untuk Meningkatkan Efektivitas Dan Efisiensi Kerja Di Bidang Pendapatandinas Perdagangan Kota Surakarta	Teknik analisis data deskriptif komparatif	Terdapat peningkatan efektivitas dan efisiensi kerja dengan menerapkan sistem otomatisasi administrasi pada pegawai Bidang Pendapatan, Dinas Perdagangan Kota Surakarta. Hal tersebut terefleksi dari beberapa indikator yaitu, kualitas kerja, produktivitas kerja, kemampuan adaptasi, kepuasan kerja, tingkat

				ekonomis dan prosedur kerja
2	Anisah dan Nia Novianti (2017)	Analisa dan Perancangan Sistem Informasi Administrasi Kepegawaian Pada Dinas Kebudayaan dan Pariwisata Provinsi Kepulauan Bangka Belitung	Metodelogi berorientasi objek yaitu menggunakan diagram Unified Modelling Language (UML).	Pada penelitian tersebut sebuah sistem terkomputerisasi mampu meminimalisir tingkat kesalahan yang terjadi pada sistem manual dan sistem dapat menjadi lebih optimal dalam aktualitas dan keakuratan dengan menggunakan konsep database.
3	Putri Kurnia Handayani (2016)	Sistem Informasi Administrasi Data Kepegawaian Pada Bagian Personalia Pt. Xyz	Perancangan sistem menggunakan oriented modelling language (OML)	Berdasarkan perancangan yang telah dilakukan berhasil dibuat sebuah sistem informasi administrasi kepegawaian berorientasi objek di Personalia PT. XYZ yang dapat menampung semua data yang menjadi tugas bagian personalia. Selain itu kemungkinan duplikasi data dapat diatasi dan juga membantu mempercepat kegiatan yang ada pada Personalia