BAB III
ANALISIS DAN PERANCANGAN

3. 1 Analisis

3.1.1 Analisis Sistem

Anilisis sistem dilakukan pada sistem yang sedang berjalan. Tujuan dari analisis sistem ini ialah mengetahui kelemahan serta kemampuan sistem yang ada di Rahastra CU. Mulai *software, hardware* dan pengguna. Hasil dari analisis sistem akan digunakan untuk membuat rancangan sistem yang akan diusulkan.

3.1.2 Analisis Masalah

Rahastra Credit Union (Rahastra CU) adalah sebuah lembaga koperasi simpan pinjam yang kegiatan utamanya ialah menghimpun dana dari para anggotanya serta menyalurkan kembali melalui bentuk pinjaman.Sebagai koperasi simpan pinjam,Rahastra CU tentu memiliki memiliki sistem untuk mendukung fungsinya. Namun karena jumlah cabang Rahastra CU yang lebih dari satu, maka sistem itu tidak menjadi kurang efektif. Hal ini dikarenakan sistem itu dibuat hanya untuk koperasi yang memiliki satu kantor saja. Akibatnya cabang-cabang Rahastra CU memilki sistem tersendiri.

Dari hasil pengamatan serta wawancara diketahui bahwa di Rahastra CU mempunyai kesulitan untuk melakukan integrasi data. Dikarenakan data tidak dari setiap cabang tidak terintegrasi maka terdapat beberapa kesulitan, diantaranya :

kesulitan pembuatan laporan dari setiap cabang dan transaksi anggota tidak dapat dilakukan disetiap kantor cabang Rahastra CU.

Untuk mengatasi masalah yang terjadi di Rahastra CU maka perlu dilakukan perbaikan, terutama sistem *Database.* Sehingga pada akhirnya *Database* dari setiap masing-masing kantor cabang dapat saling terintegrasi walaupun berjauhan.

3.1.3 Analisis Sistem Yang Sedang Berjalan

Analisi sistem yang sedang berjalan menjelaskan tentang keadaan sistem yang digunakan saat ini di Rahastra CU.

3.1.3.1 Analisis Perangkat Lunak Yang Sedang Berjalan

Perangkat lunak yang ada sekarang dapat dibedakan menjadi dua, yakni perangkat lunak yang terdapat pada komputer *server* dan perangkat lunak pada komputer *client*. Komputer *server* akan menyediakan data yang akan diakases olehkomputer *client.*

1. Perangkat Lunak Pada SisiServer
    a. Microsoft Windows Server 2003
    b. Borland Delphi 7
    c. *Database* Desktop
    d. Sikopdit (Aplikasi Pengelola Simpan Pinjam)
2. Perangkat Lunak Pada Sisi *Client*
    a. Microsoft Windows 7
    b. Borland Delphi 7

c. *Database* Desktop

d. Sikopdit (Aplikasi Pengelola Simpan Pinjam)

3.1.3.2 Analisis Perangkat Keras Yang Sedang Berjalan

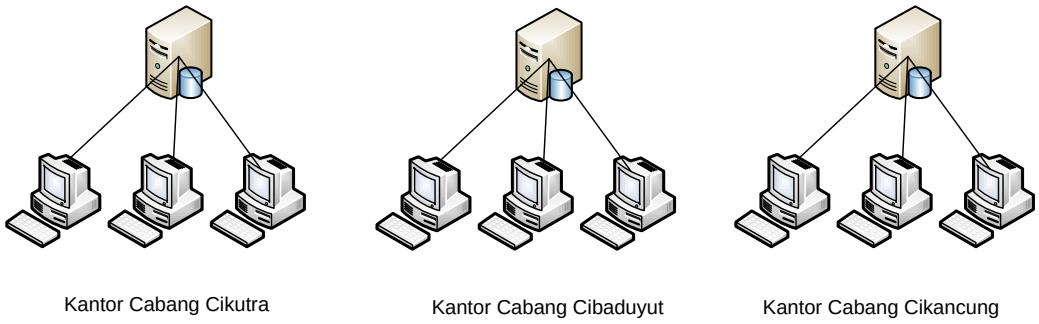
Perangkat keras yang digunakan saat ini meliputi komputer *server*, komputer *client* dan peralatan jaringan..

1. Perangkat Keras *Server*
   a. Processor : Intel® Xeon® Processor X5355 (8M Cache, 2.66 GHz, 1333 MHz FSB)
   b. Mainbord : Intel Server Board
   c. RAM : 2 Gb DDR2
   d. VGA : onboard
   e. HDD : 160 Gb
   f. Monitor : 15”
2. Perangkat Keras Komputer *Client*
   a. Processor : Intel Core 2 Duo
   b. Mainbord : Intel Maibord
   c. RAM : 1GB DDR2
   d. Monitor : 15”
   e. HDD : 160 GB
3. Peralatan Jaringan Komputer
   a. *Network Interface Card*
   b. Kabel UTP dan RJ45
   c. Modem ADSL

d. *Line* Telepon

3.1.3.3 Analisis *Database* Yang Sedang Berjalan

Berdasarkan studi kasus yang ada di Rahastra CU, maka *Database* saat ini yang digunakan menggunakan sistem terpusat, dan tidak terhubung dengan *Database* yang ada di kantor cabang lain.

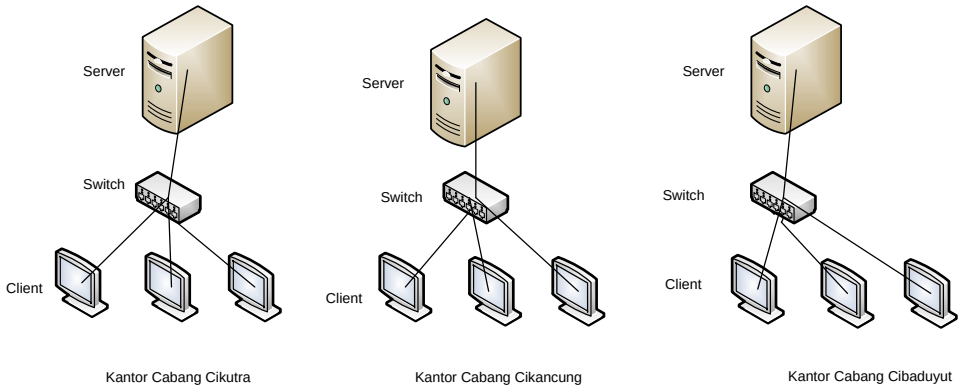


Gambar 3. 1 *Database* di Rahastra CU

Akibatnya transaksi tidak dapat dilakukan antar cabang dan pembuatan tidak dapat dilakukan secara bersamaan.

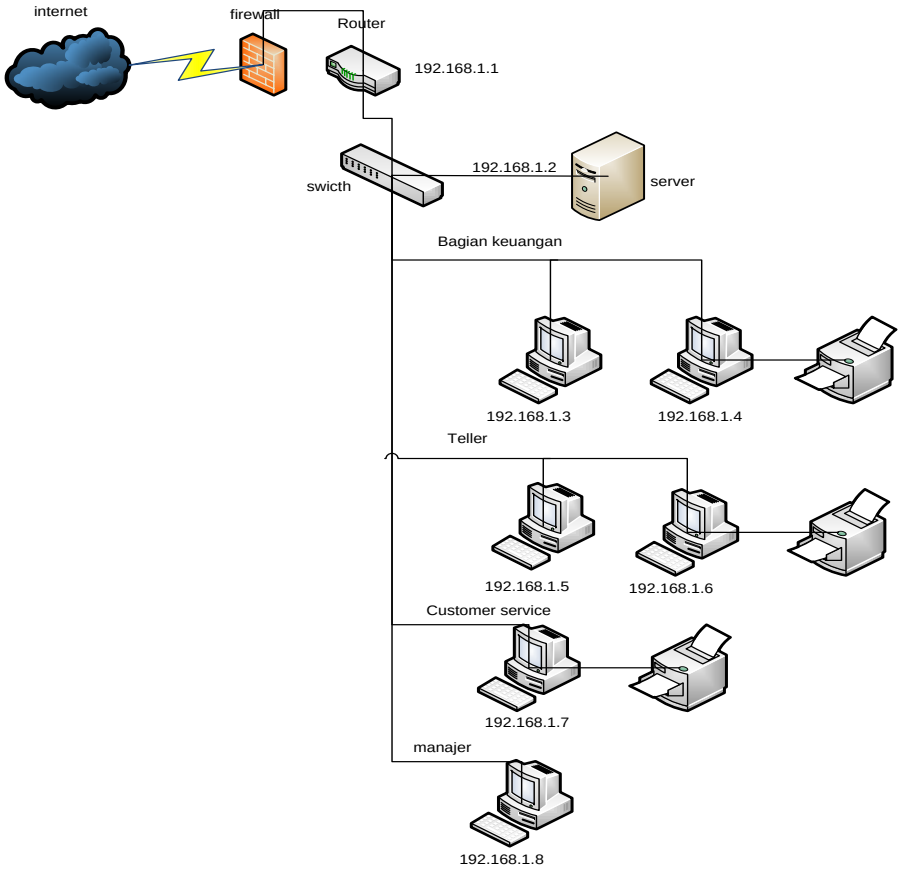
3.1.3.4 Analisis Jaringan Komputer Yang Sedang Berjalan

Pada saat ini masing-masing cabang Rahastra CU memiliki jaringan komputer sendiri-sendiri yang terpisah. Secara garis besar topologi yang digunakan oleh cabang Rahastra CU digambarkan sebagai berikut:



Gambar 3. 2 Topologi Jaringan Komputer Di Rahastra CU

Namun demikian setiap kantor cabang dan kantor pusat mempunyai arsitektur jaringan yang sama. Berikut adalah ilustrasinya:



Gambar 3. 3 Arsitektur Jaringan Kantor Cabang

3.1.3.5 Analisis Pengguna Sistem Yang Sedang Berjalan

Analisis dan kebutuhan pengguna dimaksudkan untuk mengtahui seberapa jauh pengguna sistem ini. Hal ini bertujuan untuk mengetahui karakteristik pengguna dalam mengoperasikan sistem yang telah ada, maupun sistem usulan.

Berdasarkan hasil pengamatan, pengguna yang terlibat dalam sistem yang sedang berjalan ialah:

1. Manajer
    1. Pendidikan S1 Ekonomi
    2. Mempunyai tugas untuk mengawasi dan mengluarkan kebijakan.
    3. Terbiasa menggunak
    4. an komputer dan aplikasi perkantoran serta sistem yang sedang berjalan.
2. *Teller*
    1. Pendidikan minimal D3.
    2. Mempunyai tanggung jawab melakukan dan memeriksa transaksi anggota.
    3. Terbiasa menggunakan komputer dan aplikasi perkantoran serta sistem yang sedang berjalan.
3. *Customer service*
    1. Pendidikan minimal D3.
    2. Mempunyai tanggung jawab melakukan dan memeriksa transaksi anggota.

3. Terbiasa menggunakan komputer dan aplikasi perkantoran serta sistem yang sedang berjalan.

4. Kepala Akuntansi
    1. Pendidikan Sarjana S1 Akuntansi.
    2. Mempunyai tanggung jawab melakukan dan memeriksa transaksi anggota.
    3. Terbiasa menggunakan komputer dan aplikasi perkantoran serta sistem yang sedang berjalan.

3.1.4 Analisis Kebutuhan Non-Fungsional

Analisis kebutuhan non-fungsional merupakan analisis terhadap sistem yang sudah ada si Rahastra CU serta kebutuhan terhadap sistem yang akan diusulkan.

3.1.4.1 Analisis dan Kebutuhan Perangkat Lunak

Pada sistem usulan, sistem akan menggunkan sistem terdistrbusi, maka untuk mendukung sistem ini maka diperlukan perangkat lunak tambahan, yaitu perangkat lunak yang digunakan untuk menjalankan *Virtual Private Network (VPN).*

Dimana terdapat dua buah *server* yang memiliki tugas berbeda. Pertama adalah *server* pusat, ini akan berfungsi untuk menyediakan seluruh data dari seluruh cabang. Dimana nantinya data dari semua cabang akan di simpan di *server* ini. Kedua adalah *server* cabang, ini akan berfungsi sebagai penyedia data bagi *client* yang ada dimasing-masing kantor cabang. Selain kebutuhan perangkat lunak untuk *server,* perangkat lunak juga dibutuhkan bagi komputer *client.*

1. Kebutuhan Perangkat Lunak Bagi *Server* Pusat
    1. Microsoft Windows Server 2003
    2. JDK 6
    3. MySQL Server
    4. VPN Server
2. Lunak Bagi *Server* Cabang
    5. Microsoft Windows Server 2003
    6. JDK 6
    7. MySQL Server
3. Kebutuhan Perangkat Lunak Bagi Komputer *Client*
    1. Microsoft Windows 7
    2. JDK 6

Dari hasil analisis kebutuhan perangkat lunak diatas, terdapat perbedaan antara perangkat lunak yang telah ada dan perangkat lunak yang dibutuhkan, yaitu kebutuhan akan JDK yang digunakan untuk menjalankan aplikasi sistem usulan.

3.1.4.2 Analsisis dan Kebutuhan Perangkat Keras

Kebutuhan perangkat keras bagi sistem usulan mencakup komputer *server*, komputer *client* dan peralatan jaringan.

1. Perangkat Keras *Server*
    a. Processor : Intel® Xeon® Processor X5355 (8M Cache, 2.66 GHz, 1333 MHz FSB)
    b. Mainbord : Intel Server Board

c. RAM : 2 Gb DDR2

d. VGA : onboard

e. HDD : 160 Gb

f. Monitor : 15”

2. Perangkat Keras Komputer *Client*

f. Processor : Intel Core 2 Duo

g. Mainbord : Intel Maibord

h. RAM : 1GB DDR2

i. Monitor : 15”

j. HDD : 160 GB

3. Peralatan Jaringan Komputer

e. *Network Interface Card*

f. Kabel UTP dan RJ45

g. Modem ADSL

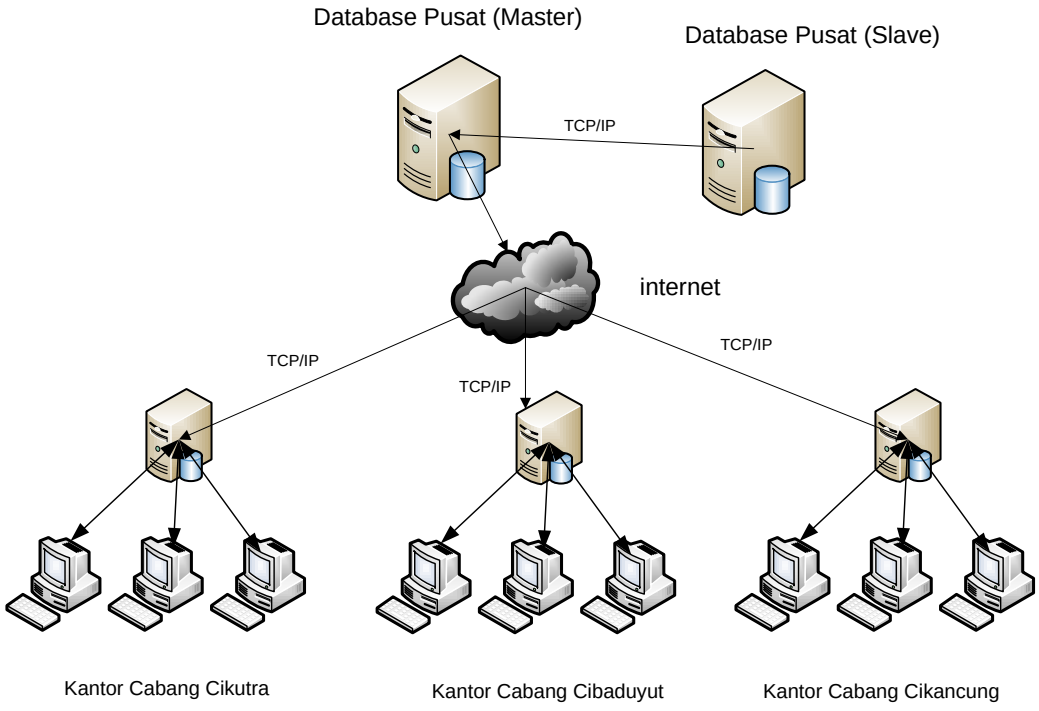
h. *Line* Telepon

Dari hasil analisis kebutuhan perangkat keras diatas tidak terdapat perbedaan antara perangkat keras yang telah ada dan perangkat keras yang dibutuhkan.

3.1.4.3 Analisis dan Kebutuhan *Database*

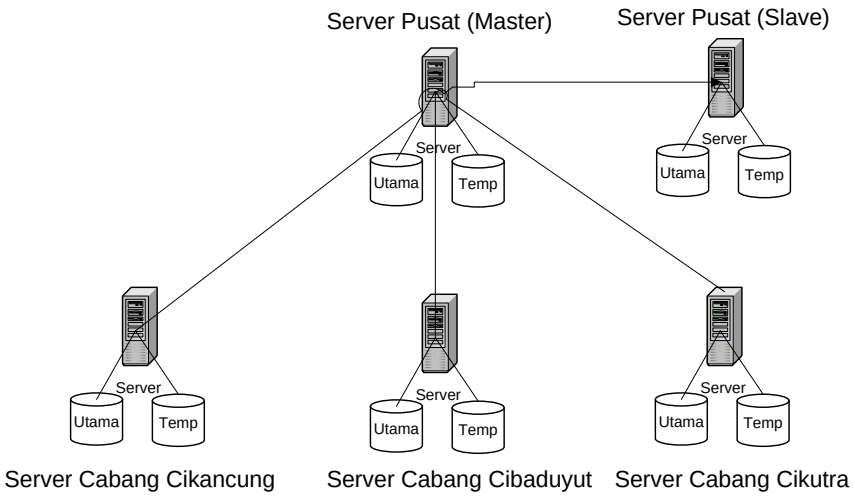
Untuk mengatasi masalah yang terjadi pada sistem *Database* yang sedang berjalan maka *Database* akan diubah menjadi terdistribusi. Diamana dengan metode ini nantinya *Database* dari setiap cabang akan disimpan dalam *Database*

pusat namun kantor cabang tetap memiliki *Database* dari cabang mereka masing-masing. Data yang terdapat di *Database* pusat akan selalu di *update* secara berkala agar datanya konisten dan relevan. Berikut adalah konsep dari metode *Database* terdistribusi:



Gambar 3. 4 Konsep *Database* Terdistribusi

Disetiap kantor cabang dan kantor pusat memiliki 2 *Database* yaitu, *Database* utama dan *Database* temporer. Setiap data transaksi akan disimpan di *Database* utama dan *Database* temporer. Nantinya *Database* utama akan menyimpan data dari keseluruhan transaksi dan *Database* temporer menyimpan data transaksi baru.



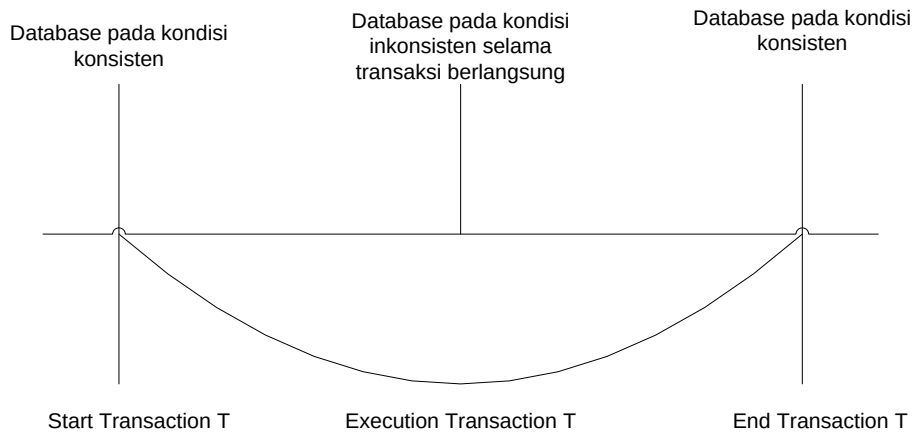
Gambar 3. 5 Arsitektur *Database* Sistem Usulan

1. Manajemen Transaksi

Transaksi ialah proses bertahap untuk membaca atau menulis data di dalam *Database.* Transaksi didalam sistem usulan dilakukan di kantor cabang. Ketika melakukan transaksi maka transaksi itu akan dijalankan pada dua *Database,* yakni *Database* utama dan *Database* temporer.

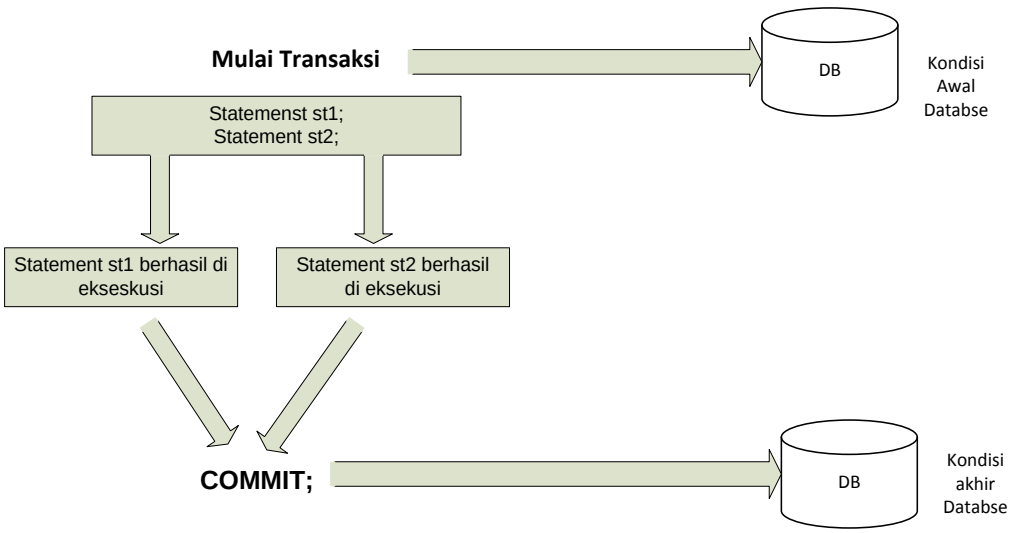
Database utama akan berfungsi sebagai penyimpanan utama sedangkan *Database* temporer akan berfungsi sebagai penyimpanan temporer yang akan digunakan untuk proses sinkronisasi.

Manajemen transaksi dilakukan untuk memastikan *Database* tetap konsisten. *Database* akan mengalami perubahan pada waktu terjadi modifikasi data. *Database* juga akan berada pada kondisi tidak konsisten selama proses modifikasi data dan akan menjadi konsisten kembali pada saat transaksi selesai.

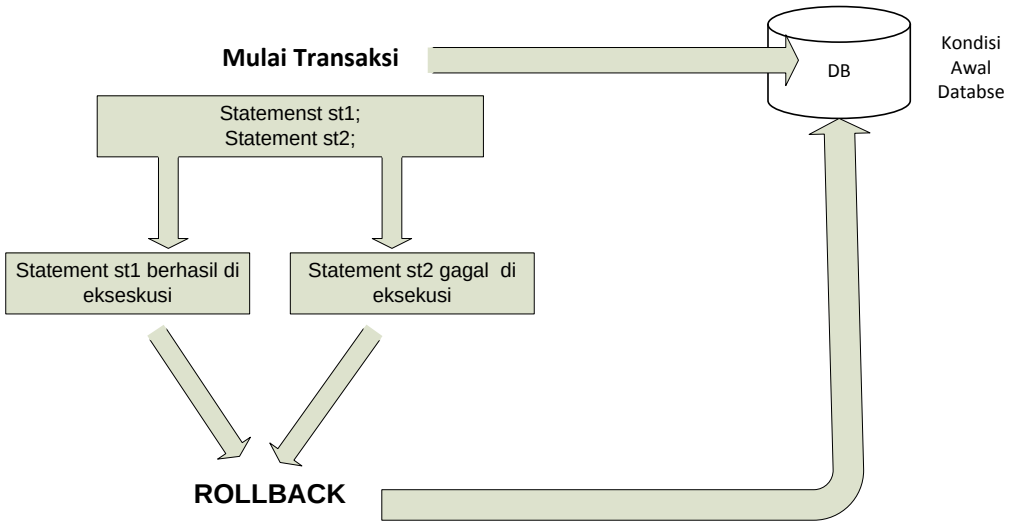


Gambar 3. 6 Model Transaksi

Transaksi dikatakan sukses jika transaksi transaksi berhasil dieksekusi tanpa ada kegagalan. Transaksi yang sukses akan disimpan secara permanen *(commit).* Namun jika transaksi tidak berhasil diekseksekusi maka transaksi dikatakan gagal. Transaksi yang gagal akan menghasilkan aksi yang belum diselesaikan *(undone action)* dan *Database* akan dikembalikan pada kondisi sebelum transaksi gagal dilaksanakan *(rollback).*



Gambar 3. 7 Transaksi Berhasil



Gambar 3. 8 Transaksi Gagal

Ada beberapa aspek untuk menjaga setiap transkasi konsisten dan *reliable*, berikut adalah kriterianya:

1. *Atomic*

   Setiap *statement* dalam sebuah transaksi dianggap sebagai satu unit. Dan hanya jika seluruh operasi transksi berhasil dilakukan maka akan diaplikasikan kedalam *Database*. Jika salah satu operasi transaksi gagal dilaksanakan maka operasi tersebut tidak akan diterapkan kedalam *Database*.

2. Konsisten

   Setiap transaksi yang dilakukan hanya memastikan data yang ada di *Database* tetap konsisten. Jika beberapa *statement* yang dilakukan salah satunya mengalami kegagalan, maka data yang ada di *Database* menjadi tidak konsisten. Dengan kata lain setiap

transaksi tidak akan mempengaruhi *Database* yang memengaruhi ketidak kosistenan data dalam *Database.*

3. Isolasi

   Setiap transaksi harus diisolasi dari transaksi lain, sehingga tidak ada *user* yang dapat mengakses data ketika sebuah transaksi dilakukan yang dapat mempengaruhi ketidak konsistenan data.

4. *Durable*

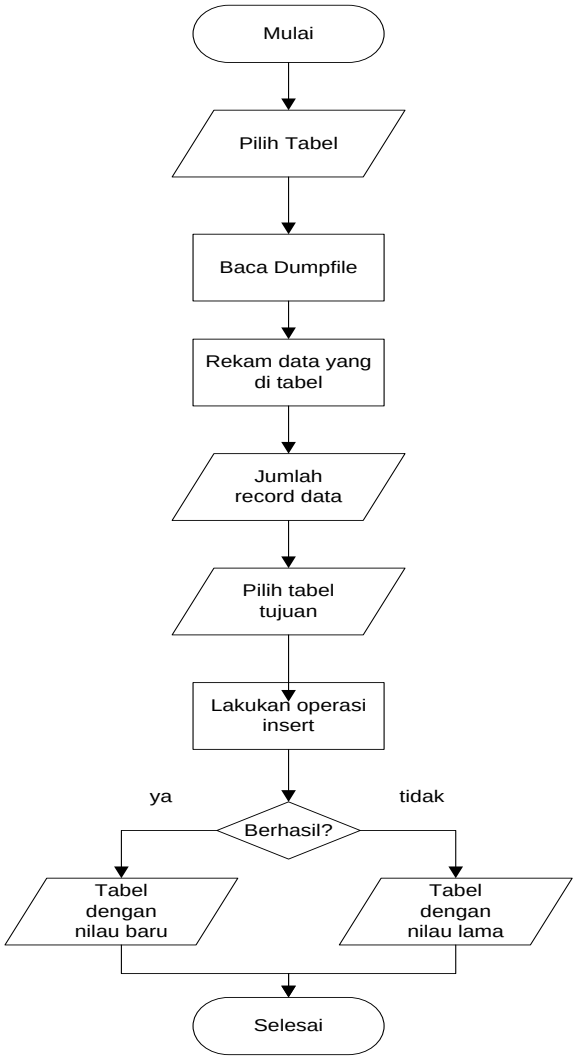
   Setiap transaksi yang berhasil dilakukan yang dapat mengakibatkan perubahan terhadap *Database*harus disimpan secara permanen meskipun terjadi *error* terhadap *hardware* maupun *software.*

2. Distribusi Data

Distribusi data dilakukan untuk mejaga konsistensi *Database*, baik itu *Database*di kantor cabang maupun kantor pusat. Distribusi data dilakukan dengan cara mengsinkronisasikan data dari *Database* cabang ke *Database* pusat ataupun sebaliknya.

Ada beberapa cara yang dapat dilakukan untuk melakukan sinkronisasi *Database* antar *Database* cabang dan dan *Database* pusat.

1. Melakukan *backup* data dari *Database* cabang kemudian melakukan peng-*copy*-an ke *Database* pusat. Cara ini tidaklah efektif karena proses melakukan *backup* data harus dilakukan secara berulang secara periodik. Kemudian meng-*copy* kan nya ke *Database* pusat.



Gambar 3. 9 Alur Sinkronisasi Dengan Metode Backup Data

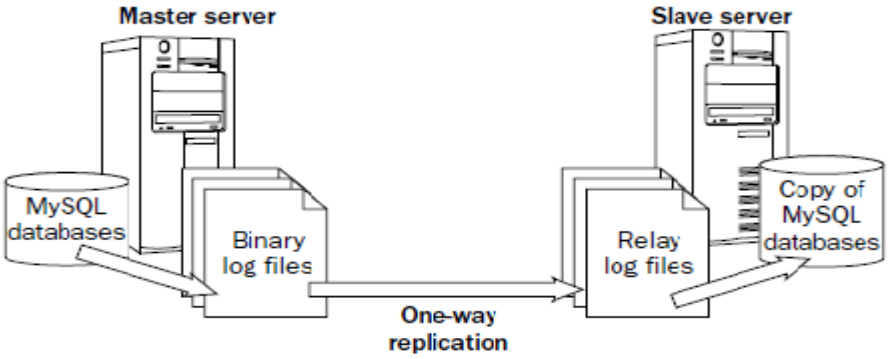
Pendistribusian ini akan menimbulkan masalah diantaranya,

a. Data yang di *backup* dari *Database* cabang adalah seluruh data yang terdapat di cabang tersebut pada saat itu yang dimungkinkan juga sudah terdapat data yang sama di *Database* pusat.

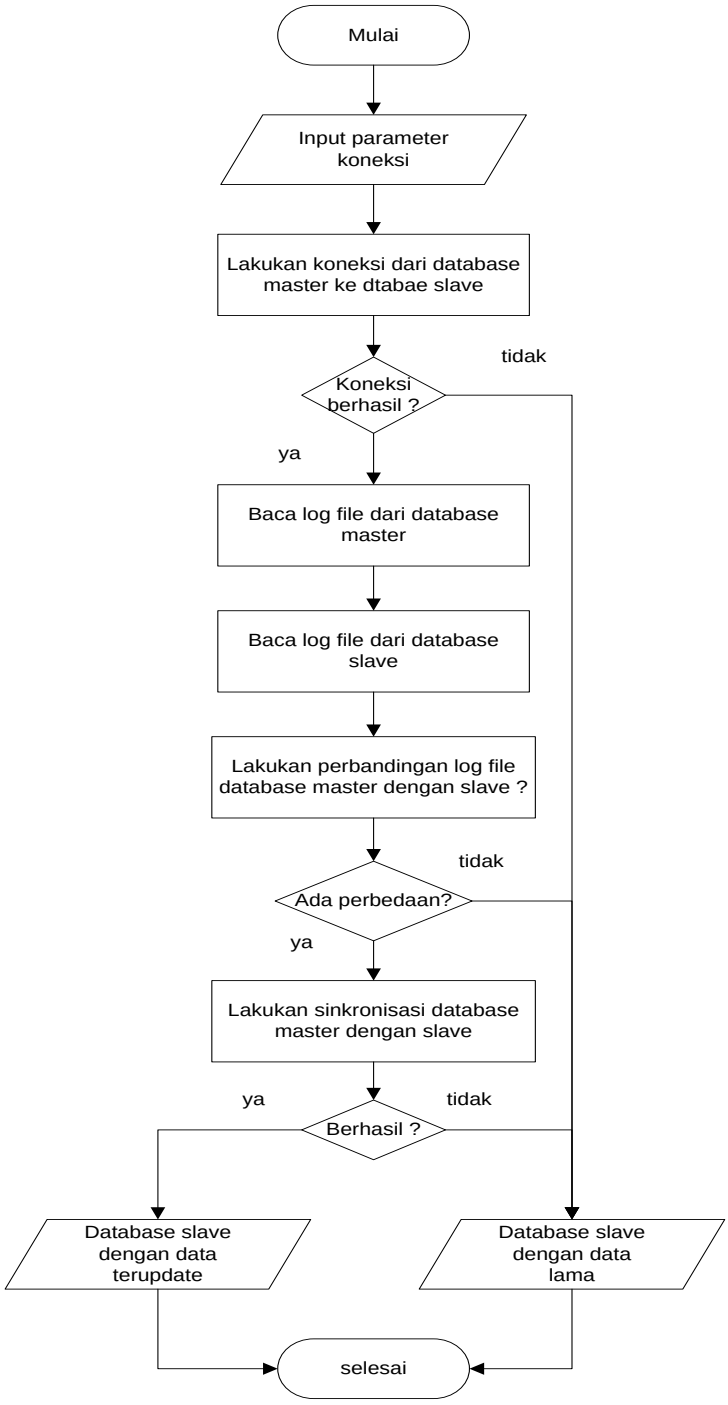
b. Data yang di *copy* kan pada *Database* pusat mungkin saja sudah tidak valid lagi. Karena proses peng-*copy*-an akan menimpa data yang sama.

c. Membutuhkan prosedur yang lebih komplek bagi seorang administrator untuk melakukan sinkronisasi data seperti ini.

2. Melakukan peplikasi *Database* menggunakan fasilitas bawaan MySQL. Saat ini MySQL telah mendukung proses replikasi. Namun replikasi yang didukung oleh MySQL adalah replikasi satu arah. Artinya MySQL hanya akan melakukan replikasi data dari komputer *master* ke komputer *slave.* Pada penelitian kali ini replikasi yang diinginkan adalah replikasi dua arah sehingga motede replikasi bawaan MySQL tidak dapat diaplikasikan.



Gambar 3. 10 Metode Relikasi MySQL



Gambar 3. 11 Alur Replikasi MySQL

Dikarenakan kedua metode diatas tidak sesuai dengan objek penelitian, maka solusinya adalah membuat aplikasi sinkronisasi *database* yang dapat melakukan sinkronisasi data dari *Database* cabang ke *database* pusat.

3. Sinkronisasi *Database*

Sinkronisasi adalah proses untuk mendistribusikan data dari kantor cabang ke kantor pusat ataupun sebaliknya dari kantor pusat ke kantor cabang.

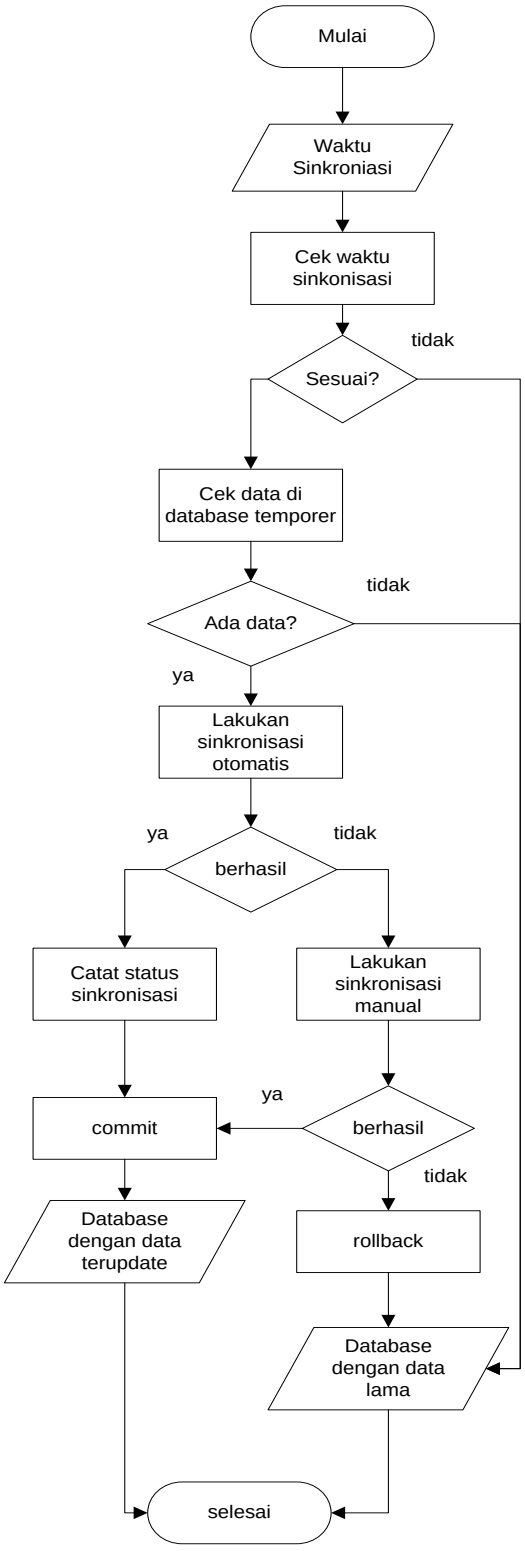
Sinkronisasi pada aplikasi ini menggunakan metode cek poin. Dimana sistem akan melakukan pengecekan kedalam *Database* temporer. Jika terdapat data, maka proses sinkronisasi akan dilakukan. Sedangakan proses sinkronisasi ini akan menggunakan metode *asynchronous* dimana data akan disinkronisasi secara berkala atau pada waktu yang telah ditentukan.

Sinkronisasi dilakukan secara otomatis, yakni pada pukul 16.00, dimana setiap kantor cabang Rahastra CU tutup. Namun demikian sinkronisasi dapat dilakukan secara manual, hal ini untuk mengantisipasi jika proses sinkronisasi secara otomatis gagal dilakukan.

Berikut adalah alur dari sinkronisasi :

1. Sistem akan memeriksa apakah terdapat data untuk di sinkronisasikan ke dalam *Database* temporer.
2. Jika tidak maka proses sinkronisasi tidak akan dilakukan.
3. Jika terdapat data, maka sistem akan melakukan sinkronisasi secara otomatis.
4. Sinkronisasi akan benar-benar disimpan jika seluruh data dalam satu tabel berhasil dikakukan sinkronisasi.

5. Jika proses sinkronisasi otomatis gagal, maka user dapat melakukan sinkronisasi secara manual yakni mengeksekusi program sinkronisasi secara manual.
6. Jika dalam proses sinkronisasi baik otomatis ataupun manual terjadi kegagalan, maka data akan di *rollback* .
7. Data yang tidak berhasil di sinkronisasikan dapat disinkronisasikan ulang, baik itu secara otomatis maupun manual.
8. Sinkroniasi yang dilakukan akan disimpan dalam tabel histori sinkroniasi.

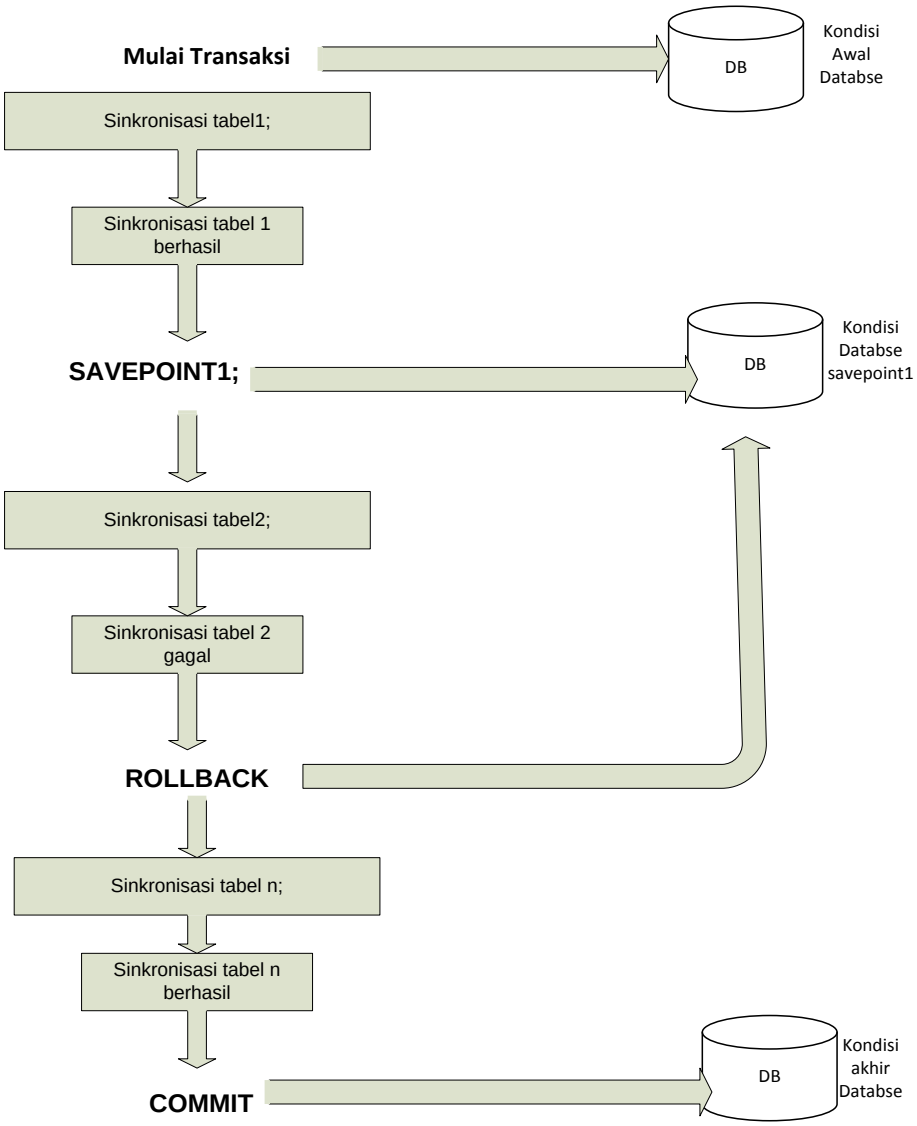


Gambar 3. 12 Alur Sinkronisasi

Error Handling Sinkronisasi

Sinkronisasi akan melibatkan banyak operasi transaksi, sehingga kemungkinan terjadi kegagalan saat proses sinkronisasi sangat beasar. Untuk menghindari ketidak konsistenan data pada saat terjadi kegagalan proses sinkronisasi maka dilakukan skenario:

1. Sinkronisasi dalam satu tabel dianggap sebagai satu unit sinkronisasi. Jadi jika ada data yang mengalami kegagalan dalam satu tabel maka seluruh proses sinkronisasi dalam tabel tersebut dibatalkan.
2. *Savepoint* akan dibuat ketika satu tabel berhasil dilakukan sinkronisasi.
3. Jika terjadi kegagalan pada sinkronosasi tabel berikutnya, maka data akan di *rollback* ka posisi *savepoint* sebelumnya.
4. Data akan benar-benar disimpan dalam *Database*jika proses sinkronisasi dalam satu unit (tabel) berhasil dilakukan.
5. Hanya sinkronisasi tabel yang gagal saja yang akan di *rollback* ke *savepoint* sebelumnya.



Gambar 3. 13 *Error Handling* Sinkronisasi

Sinkronisasi terbagi menjadi dua, sinkronisasi dari *Database*cabang ke *Database* pusat dan sinkronisasi dari *Database* pusat ke *Database* cabang

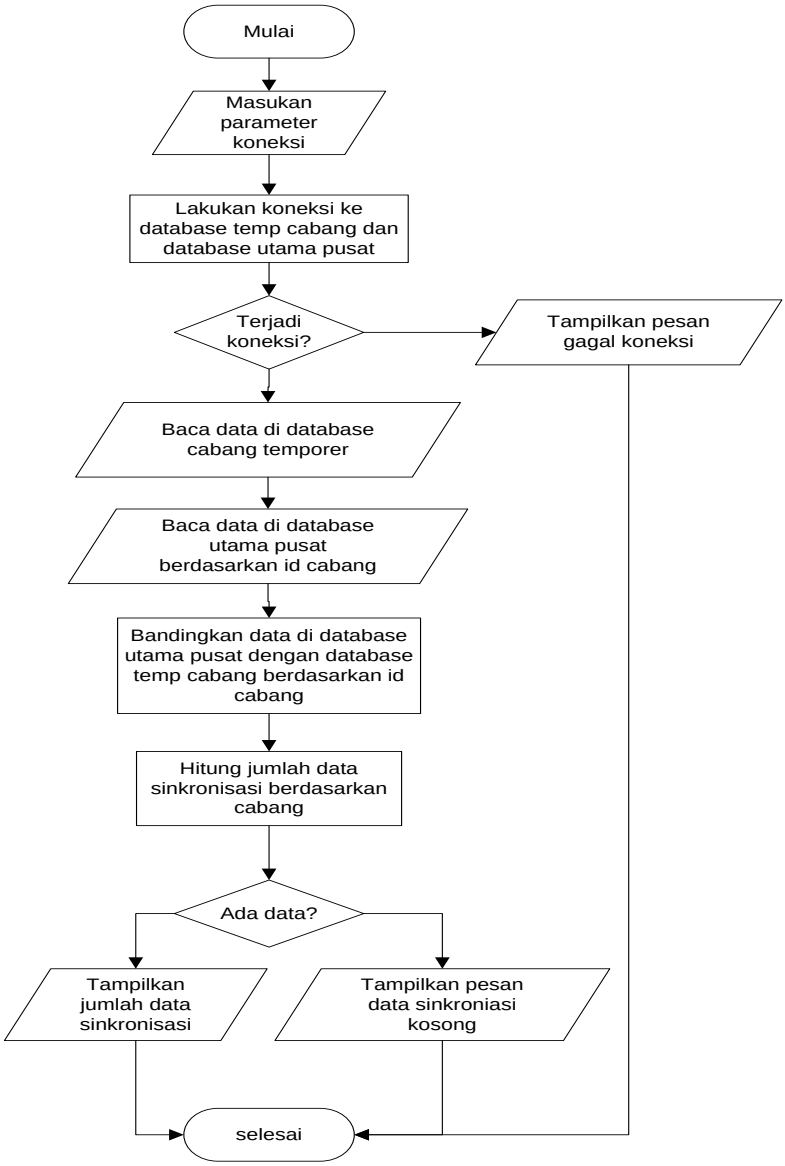
4. Proses Sinkronisasi *Database*

Untuk melakukan sinkroniasai antar dua buah *database* maka berikut adalah langkah-langkah yang dilakukan oleh sistem. Pada contoh kasus ini sistem

akan melakukan sinkronisai dari *database* kantor cabang ke *database* kantor pusat. Berikut adalah langkah-langkahnya:

1. Cek Data Sinkroniasi Ke *Database* Temporer

   Pada tahap ini, sistem akan melakukan pengecekan jumlah data yang akan disinkronisasikan. Sistem akan memeriksa jumlah data di *database* temporer cabang.



Gambar 3. 14 Alur Pengecekan Data Sinkronisasi

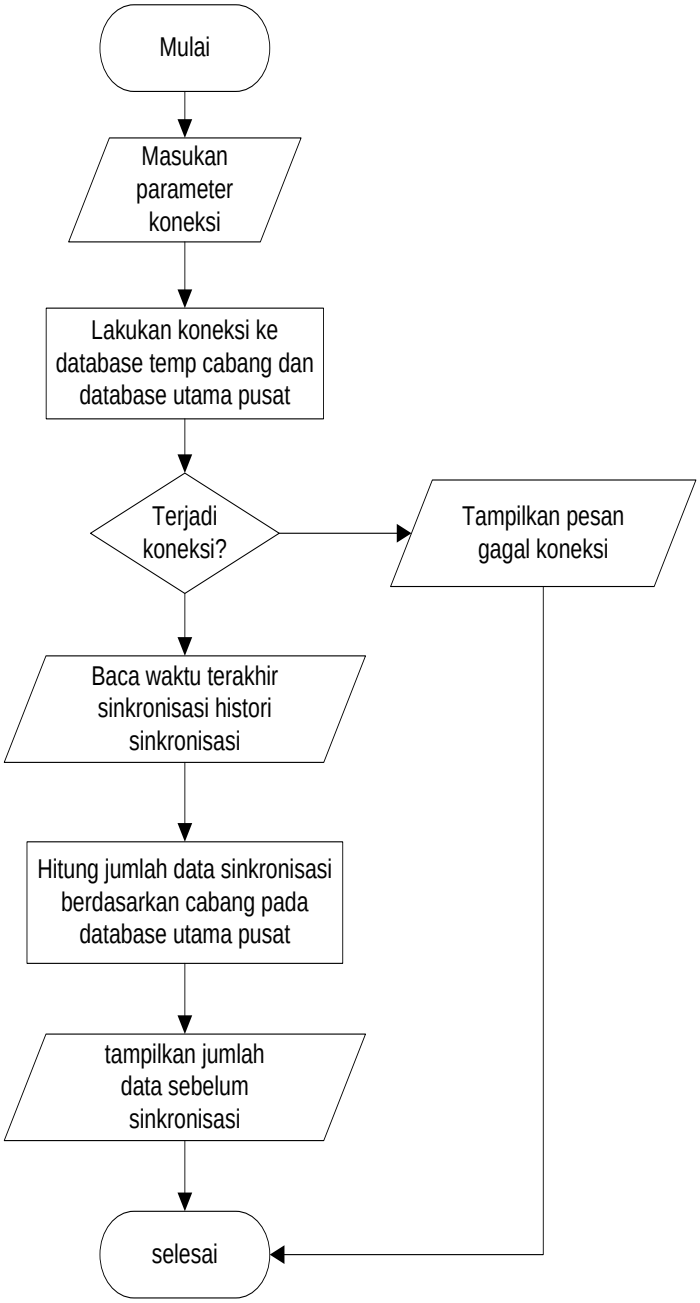
Berdasarkan proses diatas, maka didapatkan hasil sebagai berikut:

Tabel 3. 1 Hasil Pengecekan Data Sinkronisasi

Nama Tabel	Jumlah Data Sinkronisasi
Cabang	614
User	3
Rekening	614
Anggota	614
Pembukaan	614
Penutupan	0
Simpanan Saham	614
Setoran Saham	0
Penarikan Saham	0
Tabungan	0
Tabungan Anggota	0
Penarikan Tabungan	0
Setoran Tabungan	0
Pinjaman	0
Pinjaman Anggota	0
Angsuran	0
Saldo Pinjaman	0
Setoran Pinjaman	0
Penarikan Saldo Pinjaman	0
Status pinjaman	0

2. Cek Data Sebelum Sinkronisasi

Pada proses ini akan dilakukan pengecekan data di *database* utama pusat. Tujuannya adalah untuk mengetahui perubahan data yang dilakukan setelah sinkronisasi.



Gambar 3. 15 Alur Pengecekan Data Sebelum Sinkronisasi

Dari proses diatas maka akan menghasilkan data sebagai berikut:

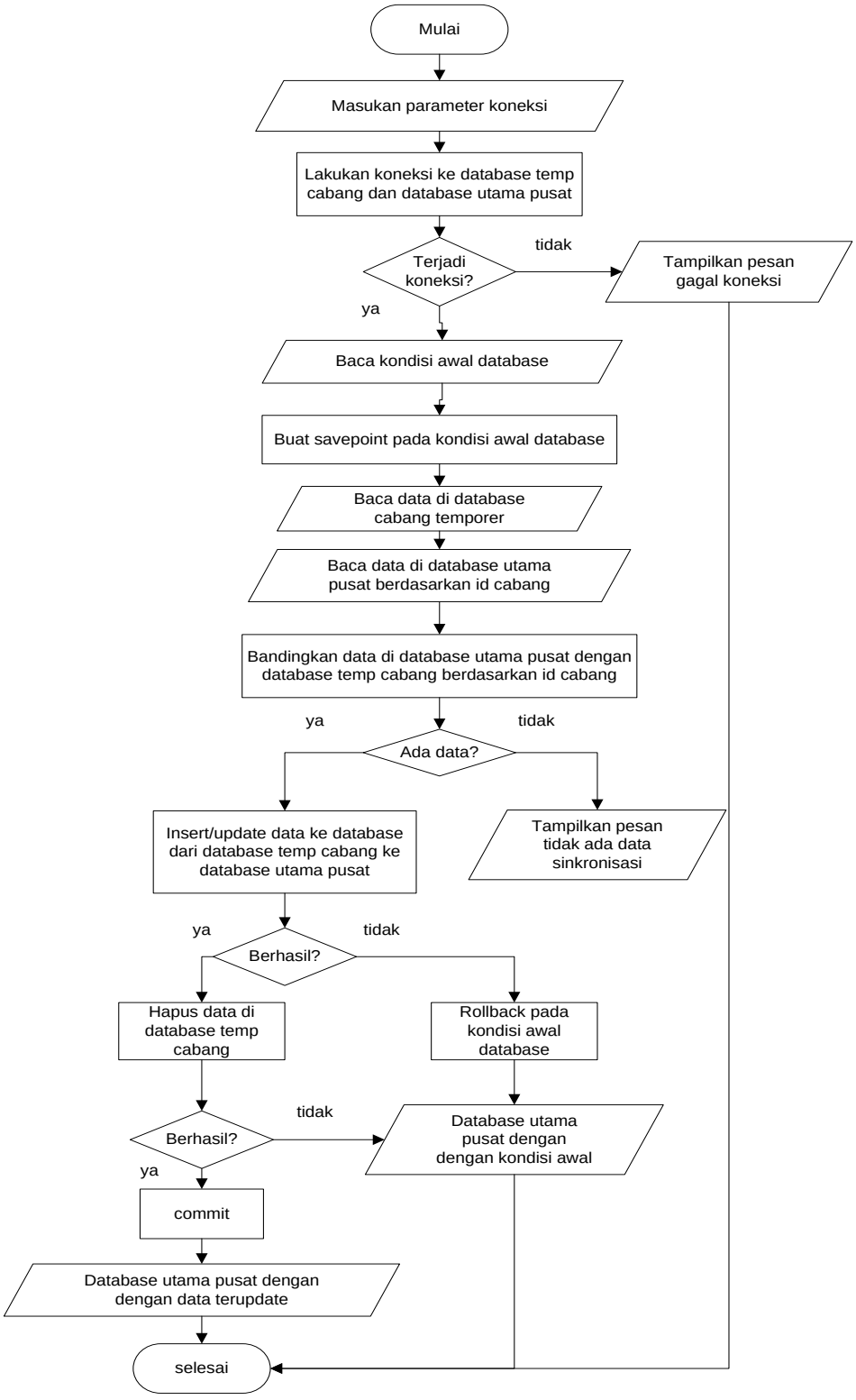
Tabel 3. 2 Hasil Pengecekan Data Sebelum Sinkronisasi

Nama Tabel	Jumlah Data Sebelum Sinkronisasi
Cabang	0
User	0
Rekening	0
Anggota	0
Pembukaan	0
Penutupan	0
Simpanan Saham	0
Setoran Saham	0
Penarikan Saham	0
Tabungan	0
Tabungan Anggota	0
Penarikan Tabungan	0
Setoran Tabungan	0
Pinjaman	0
Pinjaman Anggota	0
Angsuran	0
Saldo Pinjaman	0
Setoran Pinjaman	0
Penarikan Saldo Pinjaman	0
Status pinjaman	0

3. Melakukan sinkronisasi

   Sinkronisasi dilakukan dengan melakukan *insert/update* data dari *database* cabang temporer ke *database* utama pusat. Jika proses *insert/update* berhasil dilakukan, maka data di *database* temporer cabang akan di hapus, dan operasi sinkronisai akan di *commit* dan jika terjadi kegegalan maka akan di *rollback.*

   Berikut adalah proses sinkroniasasi:



Gambar 3. 16 Alur Sinkronisasi *Database*

Berdasarkan proses diatas maka akan menghasilkan tiga kondisi:

A. Kondisi *database* awal

Kondisi *database* awal adalah kondisi dimana *database* terakhir kali dilakukan sinkronisasi. Berikut adalah contohnya:

Tabel 3. 3 Kondisi *Database* Awal

Nama Tabel	Jumlah Data Pada *Database* Temp Cabang	Jumlah Data Pada *Database* Utama Pusat
Cabang	614	0
User	3	0
Rekening	614	0
Anggota	614	0
Pembukaan	614	0
Penutupan	0	0
Simpanan Saham	614	0
Setoran Saham	0	0
Penarikan Saham	0	0
Tabungan	0	0
Tabungan Anggota	0	0
Penarikan Tabungan	0	0
Setoran Tabungan	0	0
Pinjaman	0	0
Pinjaman Anggota	0	0
Angsuran	0	0
Saldo Pinjaman	0	0

Setoran Pinjaman	0	0
Penarikan Saldo Pinjaman	0	0
Status pinjaman	0	0

B. Kondisi ketika sinkronisasi berhasil dilakukan

Pada kondisi ini data yang ada di *database* temporer akan di *insert/update* kan ke *database* utama. Dan data yang ada di *database* temporer akan dihapus. Berikut adaah contohnya:

Tabel 3. 4 Kondisi *Database* Ketika Sinkronisasi Berhasil

Nama Tabel	Jumlah Data Pada *Database* Temp Cabang	Jumlah Data Pada *Database* Utama Pusat
Cabang	0	614
User	0	3
Rekening	0	614
Anggota	0	614
Pembukaan	0	614
Penutupan	0	0
Simpanan Saham	0	614
Setoran Saham	0	0
Penarikan Saham	0	0
Tabungan	0	0
Tabungan Anggota	0	0
Penarikan Tabungan	0	0

Setoran Tabungan	0	0
Pinjaman	0	0
Pinjaman Anggota	0	0
Angsuran	0	0
Saldo Pinjaman	0	0
Setoran Pinjaman	0	0
Penarikan Saldo Pinjaman	0	0
Status pinjaman	0	0

C. Kondisi ketika sinkronisasi gagal dilakukan.

Pada kondisi ini *database* akan dikembalikan pada kondisi dimana sinkronisasi dilakukan atau ke kondisi *database* awal. Berikut adalah contohnya:

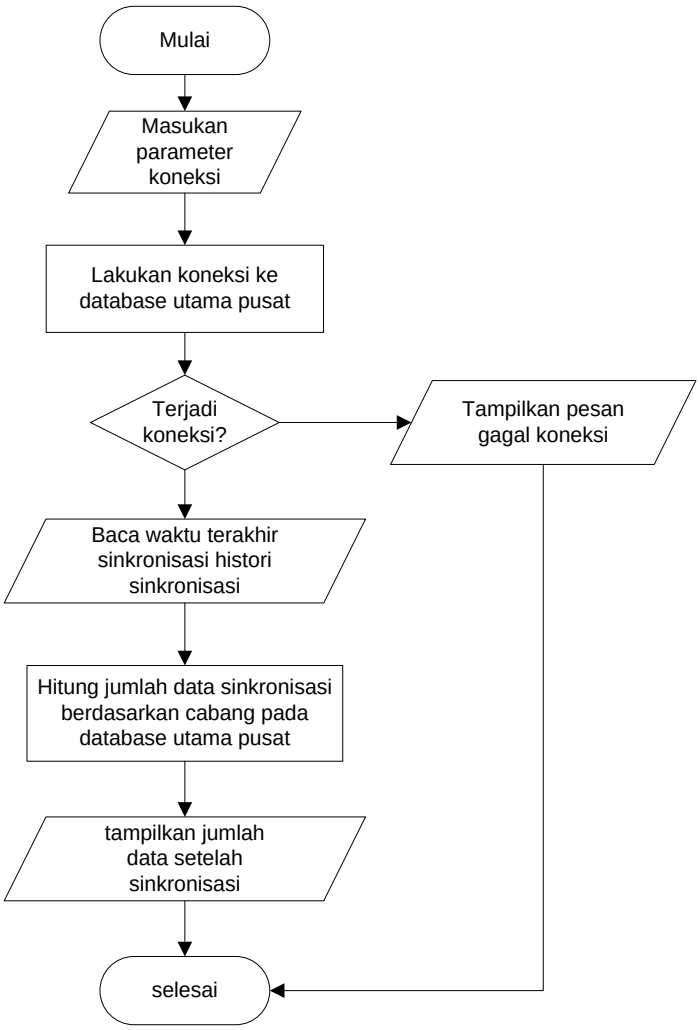
Tabel 3. 5 Urutan Sinkronisasi

Nama Tabel	Jumlah Data Pada *Database* Temp Cabang	Jumlah Data Pada *Database* Utama Pusat
Cabang	614	0
User	3	0
Rekening	614	0
Anggota	614	0
Pembukaan	614	0
Penutupan	0	0
Simpanan Saham	614	0

Setoran Saham	0	0
Penarikan Saham	0	0
Tabungan	0	0
Tabungan Anggota	0	0
Penarikan Tabungan	0	0
Setoran Tabungan	0	0
Pinjaman	0	0
Pinjaman Anggota	0	0
Angsuran	0	0
Saldo Pinjaman	0	0
Setoran Pinjaman	0	0
Penarikan Saldo Pinjaman	0	0
Status pinjaman	0	0

4. Cek Data Setelah Sinkronisasi

   Proses ini akan melakukan pengecekan data ke dalam *database* hasil sinkronisasi. Gunanya ialah untuk melihat perubahan yang terjadi setelah sinkroniasi.



Gambar 3. 17 Cek Data Setelah Sinkronisasi

Berdasarkan hasil proses diatas maka diperoleh hasil sebagai berikut:

Nama Tabel	Jumlah Data Setelah Sinkronisasi
Cabang	614
User	3
Rekening	614
Anggota	614
Pembukaan	614

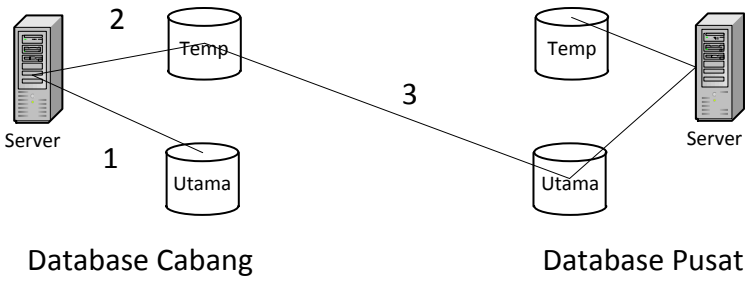
Penutupan	0
Simpanan Saham	614
Setoran Saham	0
Penarikan Saham	0
Tabungan	0
Tabungan Anggota	0
Penarikan Tabungan	0
Setoran Tabungan	0
Pinjaman	0
Pinjaman Anggota	0
Angsuran	0
Saldo Pinjaman	0
Setoran Pinjaman	0
Penarikan Saldo Pinjaman	0
Status pinjaman	0

A. Sinkronisasi *Database* Cabang Ke *Database* Pusat

Proses sinkronisasi dari *Database* cabang ke *Database* pusat dilakukan untuk mensinkronkan *Database* cabang dengan *Database* pusat. Data yang disinkronkan dari *Database* cabang ke *Database* pusat ialah seluruh data yang ada di *Database* cabang.

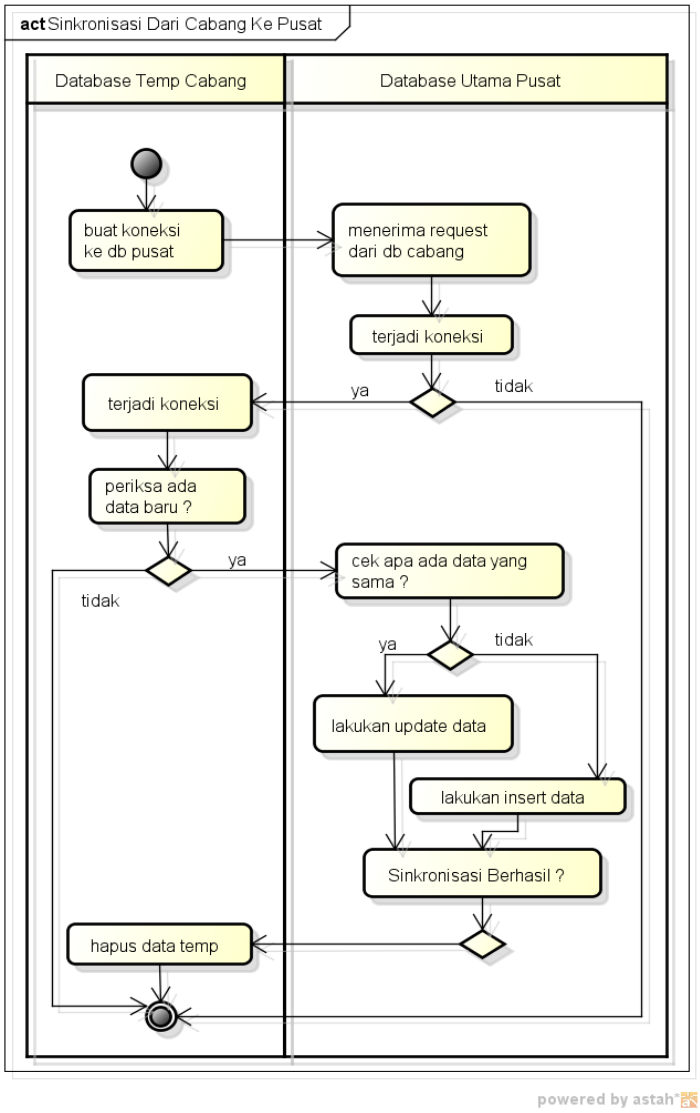
Caranya ialah, transaksi di kantor cabang akan disimpan di *Database* utama kantor cabang dan *Database* temporer kantor cabang. *Database* utama

kantor cabang akan menyimpan seluruh transaksi di kantor cabang tersebut, sedangkan *Database* temporer kantor cabang akan menyimpan data transaksi terbaru dari kantor cabang tersebut. Proses sinkronisasi dilakukan dengan cara meng-*insert* atau meng-*update* kan data dari *Database*temporer di kantor cabang ke *Database* utama dikantor pusat. Jika proses sinkronisasi berhasil maka data yang ada di *Database* temporerkantor cabang akan dihapus. Tujuannya agar data yang ada di *Database* temporer kantor cabang hanya berisi data yang belum disinkronkan dengan *Database* kantor pusat.



Gambar 3. 18 Sinkronisasi *Database* Cababng Ke *Database* Pusat

Berikut adalah prosedur sinkronisasi *Database* cabang ke *Database* pusat.



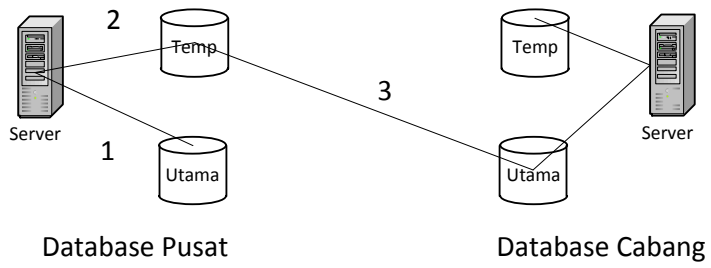
Gambar 3. 19 Prosedur Sinkronisasi *Database* Cabang Ke Pusat

B. Sinkronisasi Dari *Database* Pusat ke *Database* Cabang

Proses sinkronisasi dari *Database* pusat ke *Database* cabang dilakukan karena adanya transaksi antar cabang, sehingga dimungkinkan data dari kantor cabang dengan kantor pusat tidak lagi sesuai. Untuk menghindari masalah itu maka dilakukan sinkronisasi *Database* pusat ke *Database* cabang. Data yang disinkronkan dari *Database* pusat dengan *Database* cabang ialah data yang hanya

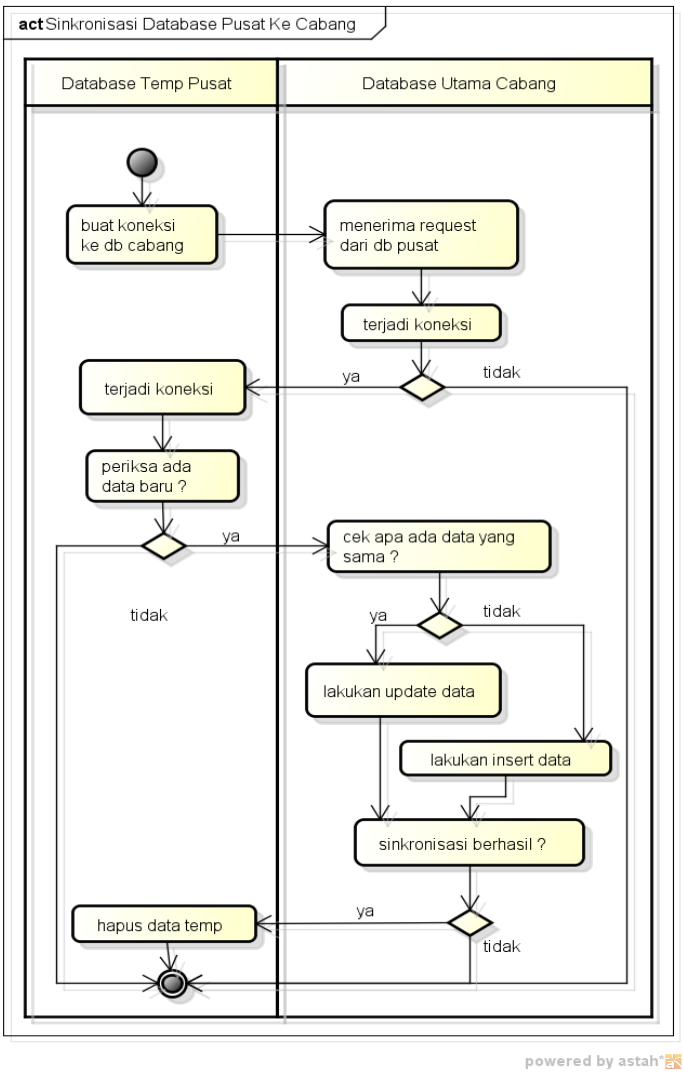
berhubungan dengan cabang tersebut saja. Dengan kata lain data cabang lain tidak akan disinkronisaikan dengan data di cabang lain.

Caranya ialah, transaksi antar cabang akan disimpan di *Database* utama pusat dan *Database* temporer pusat. Data yang ada di *Database* utama pusat ialah data keseluruhan transaksi sedangkan *Database* temporer akan menyimpan data transaski terbaru. Selanjutnya proses sinkronisasi dilakukan dengan meng-*insert* atau meng-*update* kan data dari *Database* temporer pusat ke *Database* utama pusat. Setelah proses sinkronisasi selesai maka data yang ada di *Database* temporer pusat akan dihapus. Tujuannya ialah data yang ada di *Database* temporer kantor pusat hanya berisi data yang belum di sinkronkan dengan *Database* utama kantor cabang.



Gambar 3. 20 Proses Sinkronisasi *Database* Pusat Ke Cabang

Berikut adalah prosedur sinkronisasi *Database* pusat ke *Database* cabang.

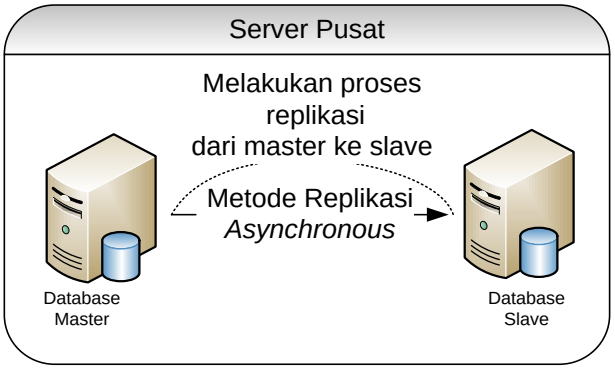


Gambar 3. 21 Prosedur Sinkronisasi *Database* Pusat Ke *Database* Cabang

5. Proses Replikasi

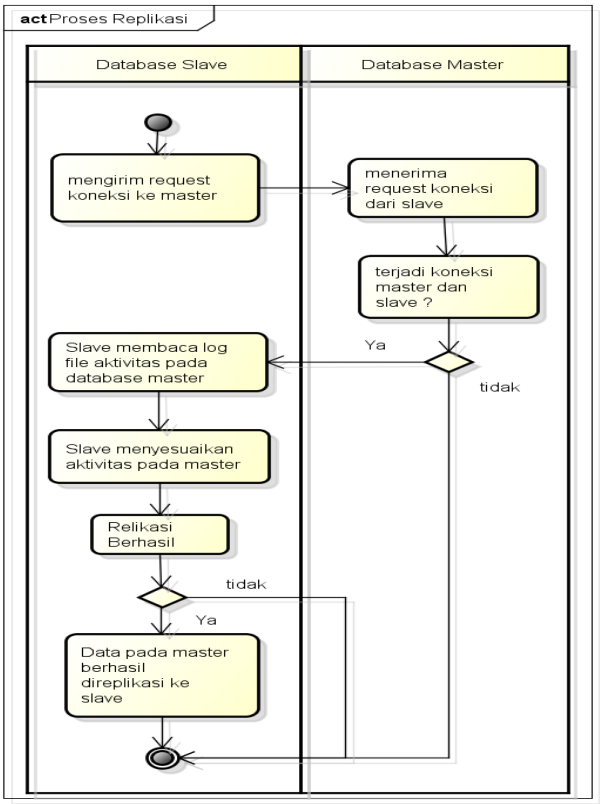
Proses replikasi dilakukan di *Database* pusat. Dimana *Database* akan direplikasi secara berkala. Hal ini bertujuan jika *Database* master mengalami gangguan maka dapat menggunakan *Database slave*. Metode replikasi yang digunakan menggunakan *asynchronous*. Dengan menggunkan metode ini

diharapkan proses replikasi dapat berjalan dengan lancar, karena replikasi dilakukan setelah proses transaksi selesai.



Gambar 3. 22 Proses Replikasi *Database*

Database pusat dianggap sebagai *Database slave*dan *Database* cabang dianggap sebagai *Databasemater.* Berikut alur proses replikasi:



Gambar 3. 23*Diagram Activity* Proses Replikasi

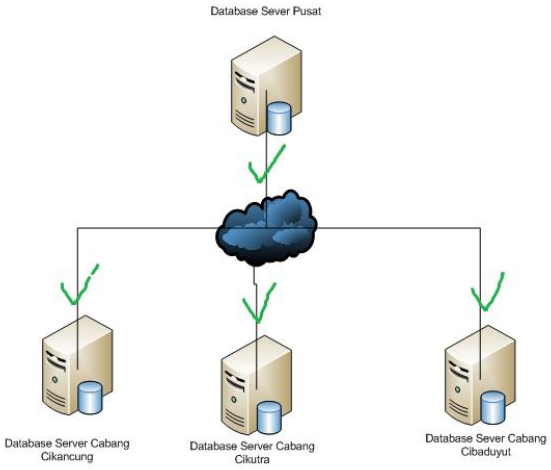
6. Koneksi *Database* Cabang dan *Database* Pusat

Proses koneksi antara *Database* cabang dengan *Database* pusat dilakukan menggunakan jaringan *public*, yaitu internet. Namun demikian koneksi ini menggunakan VPN, sehingga walaupun menggunakan jaringan *public* koneksi tetap aman.

Status Koneksi *Database*

1. *Database* cabang tekoneksi dan *Database* pusat terkoneksi

   Pada kondisi ini maka seluruh transaksi baik itu setoran maupun penarikan antar cabang maupun dalam cabang dapat dilakukan. Status ini merupakan status paling ideal yang harus dicapai.

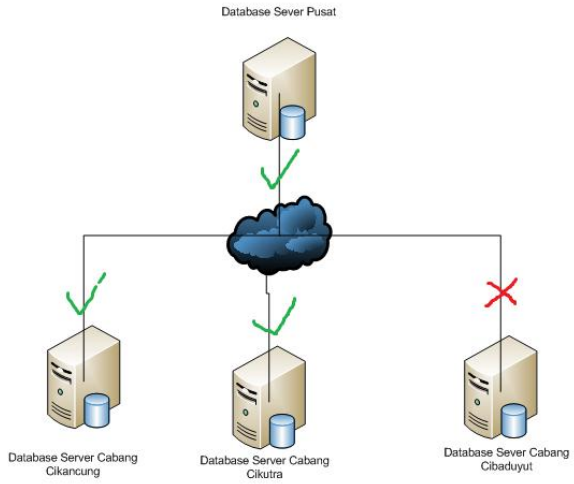


Gambar 3.24 Status *Database* Pusat dan *Database* Cabang Terkoneksi

2. *Database* salah satu cabang mati dan *Database* pusat terkoneksi.

   Pada kondisi ini transaksi tidak dapat dilakukan dimana *Database* setempat mati. Hal ini karena proses transaksi akan selalu melibatkan *Database* cabang. Untuk mengatasinya maka transaksi

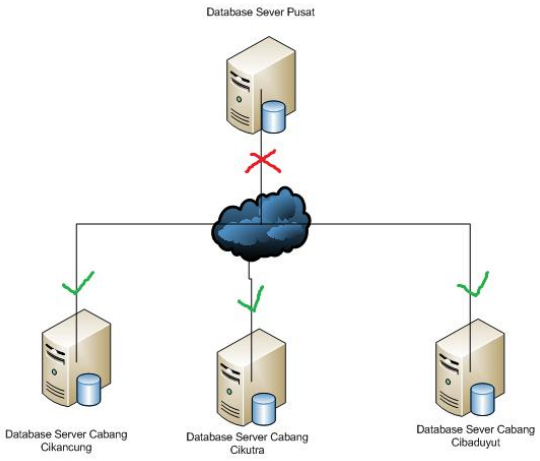
dapat dilakukan di cabang lain yang *Database* cabangnya terkoneksi.



Gambar 3.25 *Database* Pusat terkoneksi dan salah satu *Database* cabang mati

3. *Database* cabang terkoneksi dan *Database* pusat mati.

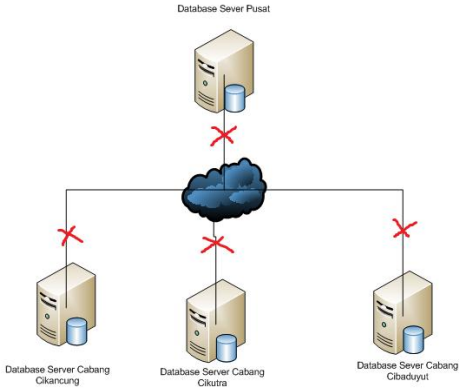
   Pada kondisi ini, transaksi hanya dapat dilakukan untuk transaksi dalam cabang saja. Transaksi antar cabang tidak dapat dilakukan.



Gambar 3.26*Database* Pusat mati dan *Database* cabang terkoneksi

4. *Database* seluruh cabang mati dan *Database* pusat mati.

   Pada kondisi ini semua tranaksi baik itu antar cabang maupun dalam cabang tidak dapat dilakukan. Kondisi ini merupakan kondisi paling buruk dan harus dihindari.



Gambar 3.27*Database* Pusat mati dan *Database* cabang mati

Dari hasil analisis *Database* diatas, terlihat perbedaan antara sistem *Database* yang sedang berjalan dengan sistem *Database* usulan. Perbedaanya ialah metode pendistribusian data.

Untuk mendukung proses pembangunan replikasi *Database* secara terdistribuis maka akan menggunakan DBMS MySQL. Ada beberpa kelebihan mengapa menggunakan RDBMS MySQL:

1. Skalabilitas

   MySQL dapat menangani *Database*dengan ukuran yang besar.

2. Portabilitas

   MySQL dapat dijalankan di berbagai sistem operasi, sepertu Linux, UNIX, Windows, Solaris dan MacOS. Selain itu MySQL dapat dijalankan di komputer *low end* sampai *mainframe.*

3. Konektifitas

   MySQL mendukung TCP/IP socket, UNIX socket dan *named pipes.* Selain itu MySQL dapat diakses dimanapun melalui internet dan dengan banyak user secara simultan.

4. Keamanan

   MySQL menyertakan sistem kontrol yang mumpuni. Sistem menggunakan struktur *host and user-based* yang dapat mengontrol informasi yang diakses dan hak akses. MySQL juga mendukung protokol *Secure Socket Layer (SSL)* untuk mendukung keneksi terenkripsi.

5. Penggunaan yang mudah

   MySQL adalah DBMS yang mudah di-*install* dan diimplementasikan.

6. *Open Source*

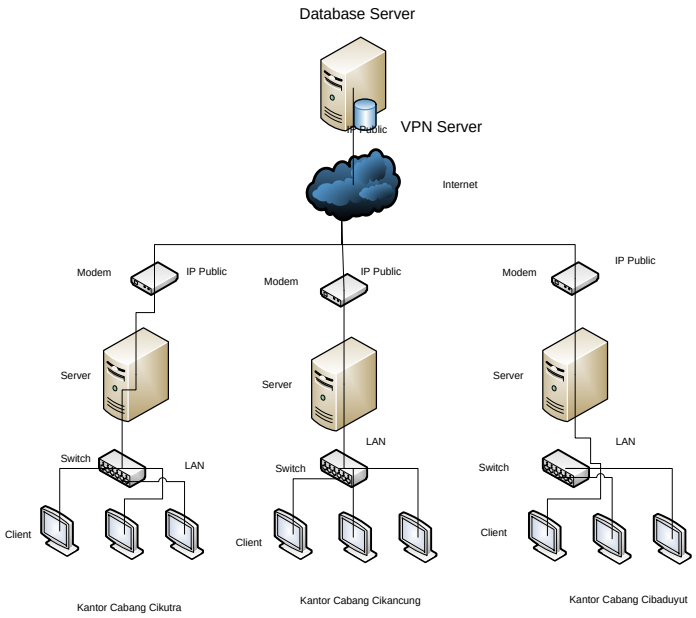
   MySQL AB membuat *source code* MySQL *open source*sehingga siapa pun yang memerlukannya dapat mendownloadnya.

3.1.4.4 Analisis dan Kebutuhan Jaringan Komputer

Dengan akan diimplementasikan *Database*terdistribusi di Rahastra CU, maka kebutuhan jaringan akan berubah, terumata pada koneksi antar cabang. Hal ini dikarenakan terjadi penambahan *server* pusat.

Database server pusat akan diakses melalui *Virtual Private Network (VPN)* sehingga untuk mengakses data pusat diperlukan koneksi internet.

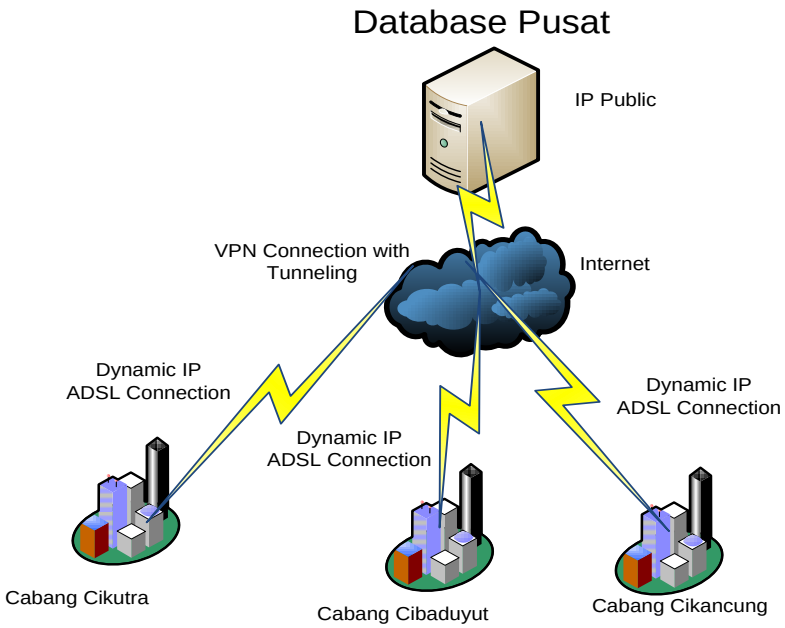
Dengan penggunaan VPN diharapkan proses pengaksesan data akan lebih aman walaupun dilakukan melalui jaringan *public* (internet).



Gambar 3. 28 Topologi Kebutuhan Jaringan Komputer Sistem Usulan

Dengan topologi jaringan diatas, maka diperlukan IPPublicuntuk setiap server. IP Publik ini diidapatkan dari penyedia jasa internet (ISP).

Topologi yang digunakan untuk membangun *Database* terdistrubis ini menggunkan toplogi *client-server*. Berikut adalah gambaran arsitektur jaringan sistem usulan.



Gambar 3. 29 Arsitektur Jaringan Sistm Usulan

Proses koneksi antara *Database* pusat dengan *Database* cabang akan menggunakan koneksi VPN. Tujuannya ialah untuk membuat koneksi antar cabang lebih aman. Berikut adalah metode pengamanan VPN:

1. Setiap terjadi konkeksi antar cabang, maka koneksi tersebut akan membuat *tunnel*. Sehingga seolah-olah koneksi tersebut terjadi dalam satu jaringan lokal.
2. Setiap pengiriman data, maka data tersebut akan di enkripsi

3. Proses koneksi hanya akan terjadi jika *client* yang mengakses server telah melalui verifikasi. Dimana *client (*kantor cabang) akan memasukan username dan password yang tlah diberikan.

Dari hasil analisis diatas terdapat perbedaan antara jaringan komputer yang telah ada di Rahastra CU dengan jaringan komputer sistem usulan, terutama dalam hal akses VPN.

3.1.4.5 Analisis dan Kebutuhan Pengguna

Untuk melakukan sinkronisasi maka diperlukan administrator disetiap kantor cabnag untuk melakukan sinkronisasi. Sedangkan administratot pusat diperlukan untuk memonitor dan meilihat data yang telah dilakukan oleh administrator cabang. Berikut adalah penjelasannya :

1 Administartor Cabang

Administrator cabang akan bertugas untuk melakukan sinkronisasi *Database* baik itu dari dari *Database* pusat ke cabang maupun sebaliknya.

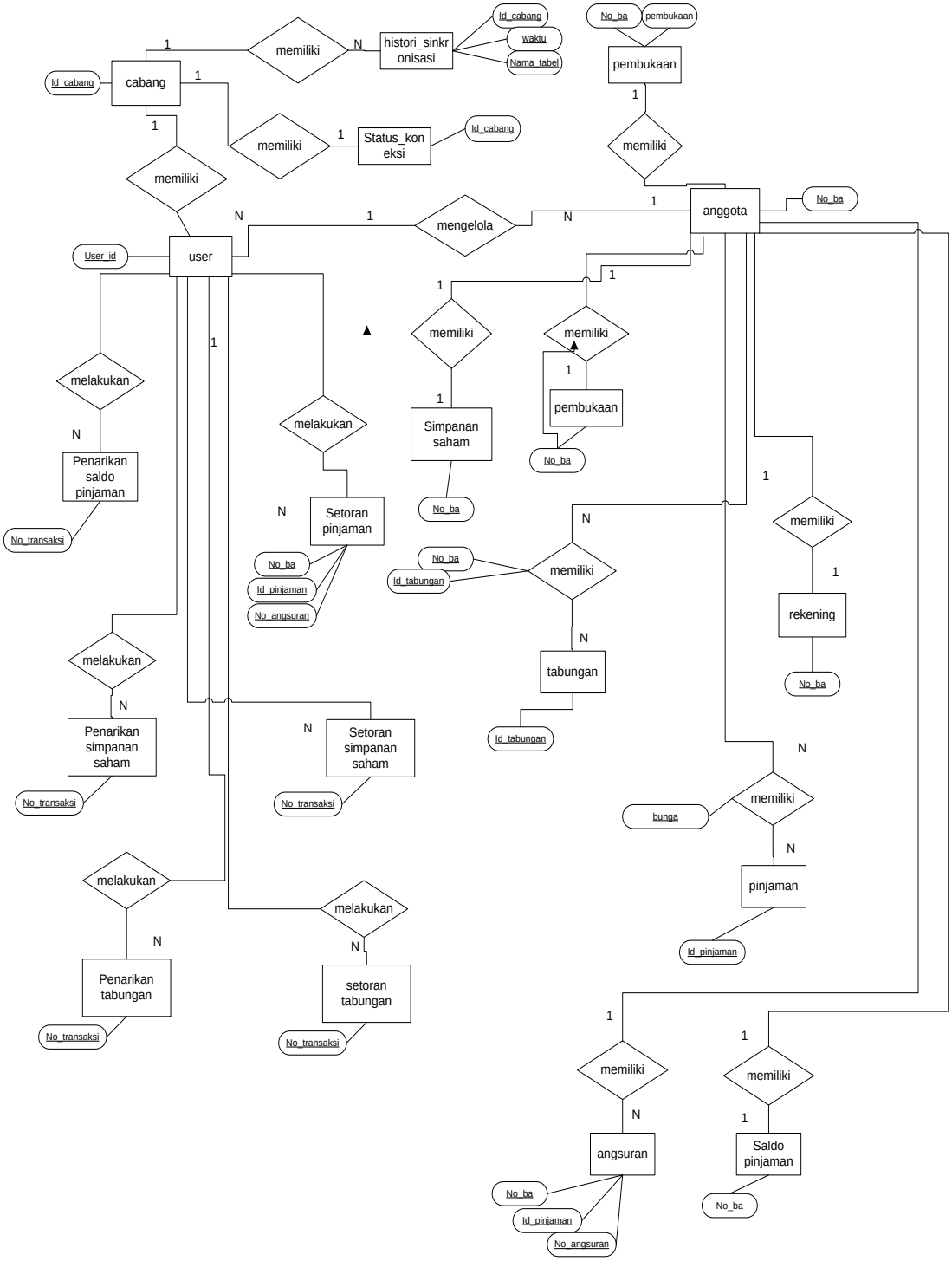
2 Administrator Pusat

Administratot pusat untuk melihat data hasil sinkronisasi dari *Database* cabang.

3.1.5 Analisis Kebutuhan Fungsional

Analisi kebutuhan fungsional akan menjelaskan tentang kebutuhan sistem usulan.

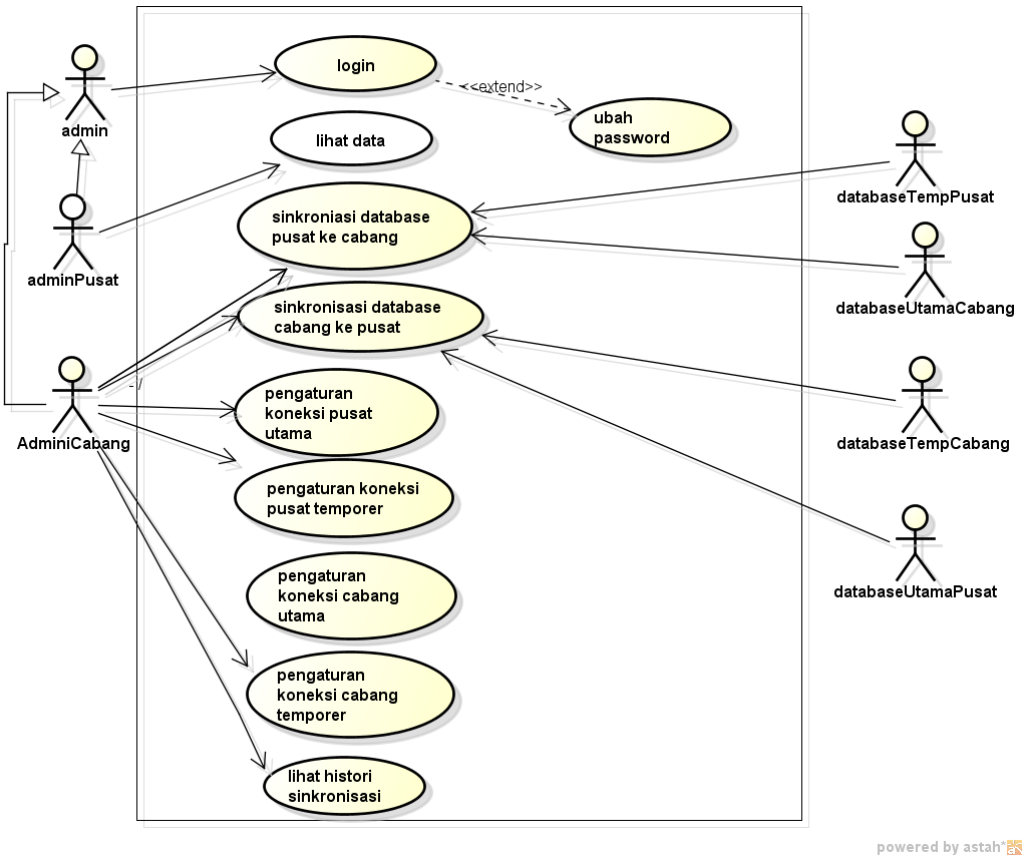
3.1.5.1 Entity *Relationship* Diagram (ERD)



Gambar 3. 30 Diagram ERD

3.1.6 *Use Case Diagram*

Use case diagram sistem yang diusulkan menggambarkan seberapa banyak aktor yang terlibar dalam sistem yang ada di kantor cabang. Diagaram ini juga menggambarkan bagiamana aktor beriteraksi dengan sistem. *Use case* juga memberikan gambaran mengenai batasan-batasan dari aksi yang dapat dilakukan oleh aktor.



Gambar 3. 31*Use Case Diagram*

3.1.6.1 Definisi Aktor

Terdapat 7 aktor yang ada dalam sistem ini. Setiap aktor mempunyai kewenangan yang berbeda-beda, berikut adalah penjelasannya :

Tabel 3. 6 Definisi Aktor Cabang

No	Aktor	Definisi
1	Admin	Administrator merupakan aktor yang akan menjalankan sistem.
2	AdminPusat	Administrator merupakan aktor yang akan menjalankan sistem pusat.
3	AdminCabang	Administrator merupakan aktor yang akan menjalankan sistem cabang
4	*database*TempPusat	Merupakan aktor ekternal yang akan menjalankan sinkronisasi dari pusat ke cabang
5	*database*UtamaCabang	Merupakan aktor ekternal yang akan menjalankan sinkronisasi dari pusat ke cabang
6	*database*TempCabang	Merupakan aktor ekternal yang akan menjalankan sinkronisasi dari cabang ke pusat
\7	*database*UtamaPusat	Merupakan aktor ekternal yang akan menjalankan sinkronisasi dari cabang ke pusat

3.1.6.2 Definisi *Use Case*

Terdapat delapan *use case* dalam sistem ini. Setiap *use case* menggambarkan operasi-operasi yang berbeda.

Tabel 3. 7 Definisi *Use Case*

No	*Use Case*	Deskripsi
1	Login	Proses bagi admin untuk

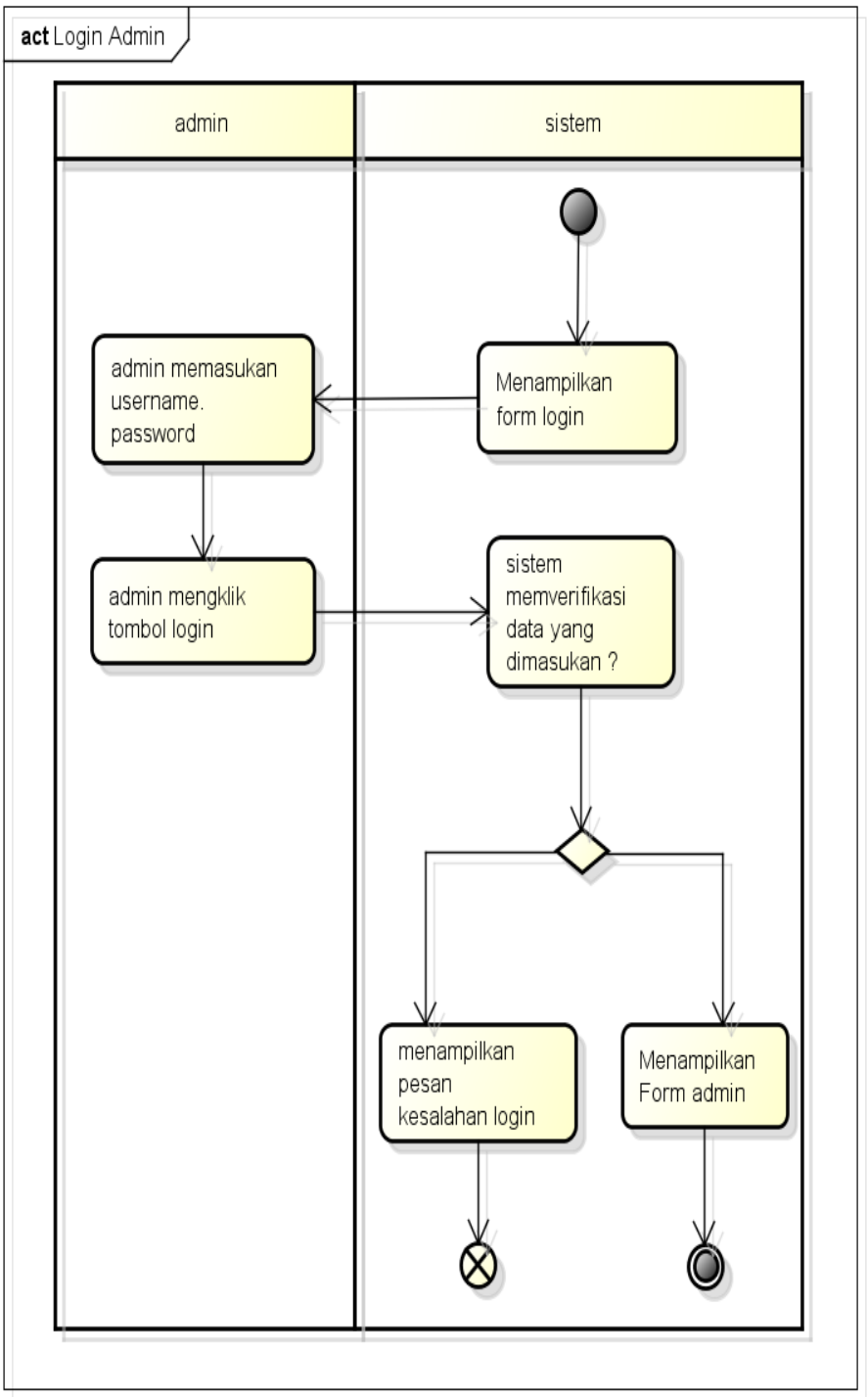
		memastikan administrator memiliki akses ke dalam sistem admin
2	Ubah password	Operasi yang digunakan admin untuk mengubah password.
3	Pengaturan Koneksi Pusat Utama	Proses untuk mengatur koneksi terhadap *Database* pusat utama
4	Pengaturan Koneksi Pusat Temporer	Proses untuk mengatur koneksi terhadap *Database* pusat temporer
5	Pengaturan Koneksi Cabang Utama	Proses untuk mengatur koneksi terhadap *Database* cabang utama
6	Pengaturan Koneksi Cabang Temporer	Proses untuk mengatur koneksi terhadap *Database* cabang temporer
7	Sinkronisasi *Database* Pusat Ke Cabang	Proses untuk melakukan sinkronisasi *Database* dari *Database* temporer pusat ke *Database* utama cabang
8	Sinkronisasi *Database* Cabang ke pusat	Proses untuk melakukan sinkronisasi *Database* dari *Database* temporer cabang ke *Database* utama pusat
9	Lihat Histori Sikronisasi	Proses untuk melihat daftar sinkronisasi yang telah dilakukan

3.1.6.3 Skenario *Use Case* Login

Proses ini melakukan autentifikasi terhadap aktor dan memastikan bawah aktor memiliki hak akses terhadapa sistem.

Tabel 3. 8 Skenario *Use Case* Login Admin

Identifikasi	
Nomor	1
Nama	Login
Tujuan	Memastikan administrator memiliki hak akses kedalam sistem
Deskripsi	Sistem akan memeriksa apakah administrator memilki hak akses untuk masuk kedalam sistem dengan memasukan password dan username.
Aktor	Administrator
Skenario Utama	
Kondisi awal	Menampilkan form login.
Aksi Aktor	**Reaksi Sistem**
1. Administrator memasukan username dan password	2. Sistem akan memverifikasi apakah data yang dimasukan telah sesuai.
	3. Jika verifikasi valid, maka sistem akan menampilkan halaman admin 4. Jika verifikasi gagal maka sistem akan menampilkan pesan kesalahan login
Kondisi akhir	Administrator dapat masuk kedalam sistem dan sisten menampilkan form admin



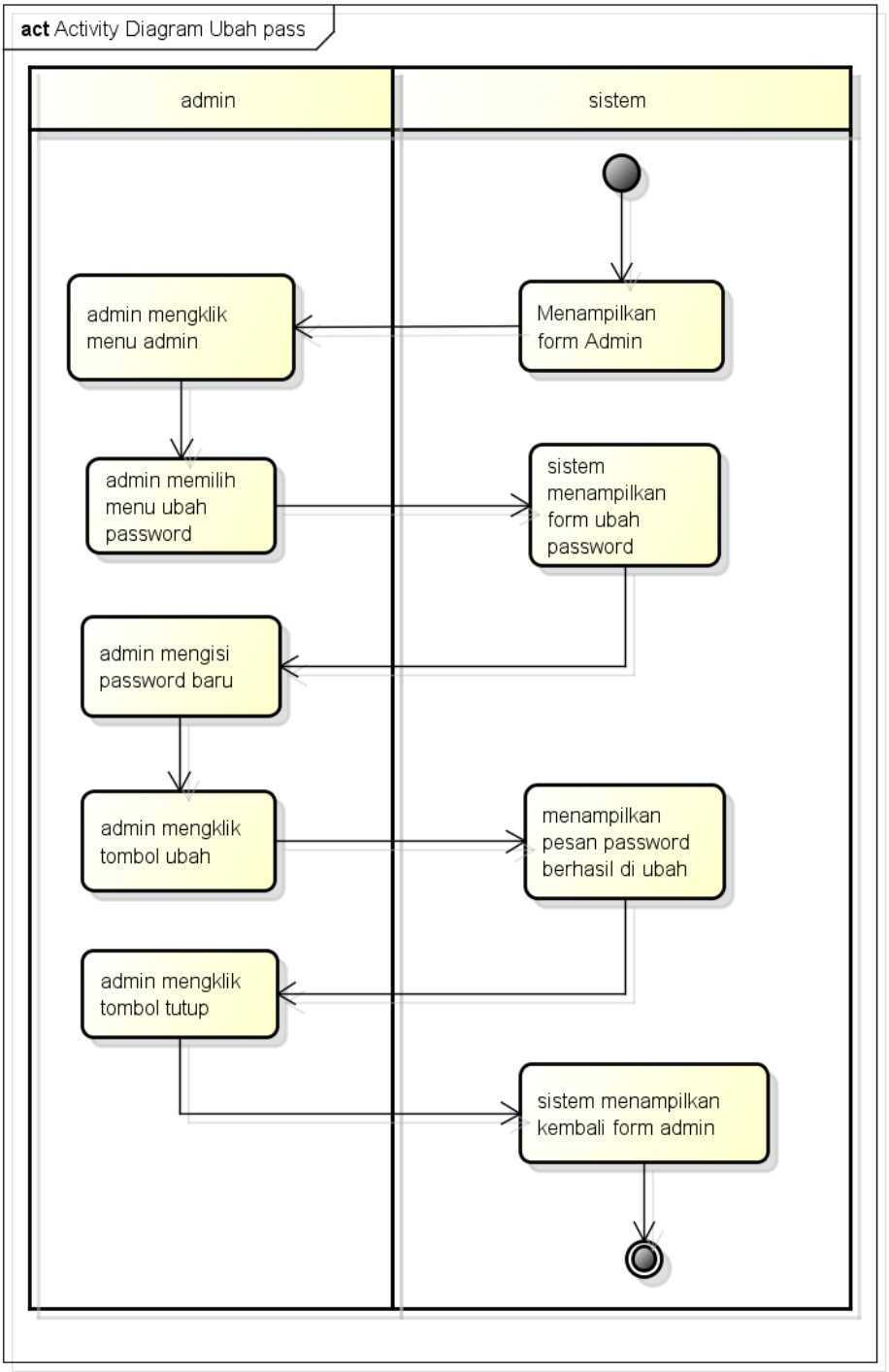
Gambar 3. 32 Activity Diagram Login

3.1.6.4 Skenario *Use Case* Ubah Password

Proses ini akan mengubah password admin. Berikut adalah penjelasannya:

Tabel 3. 9 Skenario *Use Case* Ubah Password

Identifikasi		
Nomor	2	
Nama	Ubah Password	
Tujuan	Merubah username dan password administrator	
Deskripsi	Sistem akan mengubah username dan password administrator.	
Aktor	Administrator	
Skenario Utama		
Kondisi awal	Admin sudah login, dan sistem menampilkan form ubah password	
Aksi Aktor		**Reaksi Sistem**
1. Administrator memasukan username dan password baru. 2. Admin mengklik tombol simpan. 4. Klik tombol tutup		3. Sistem akan menyimpan password dan username. 5. Menutup form ubah password dan kembali ke form admin
Kondisi akhir	Username dan password administrator sudah dirubah	



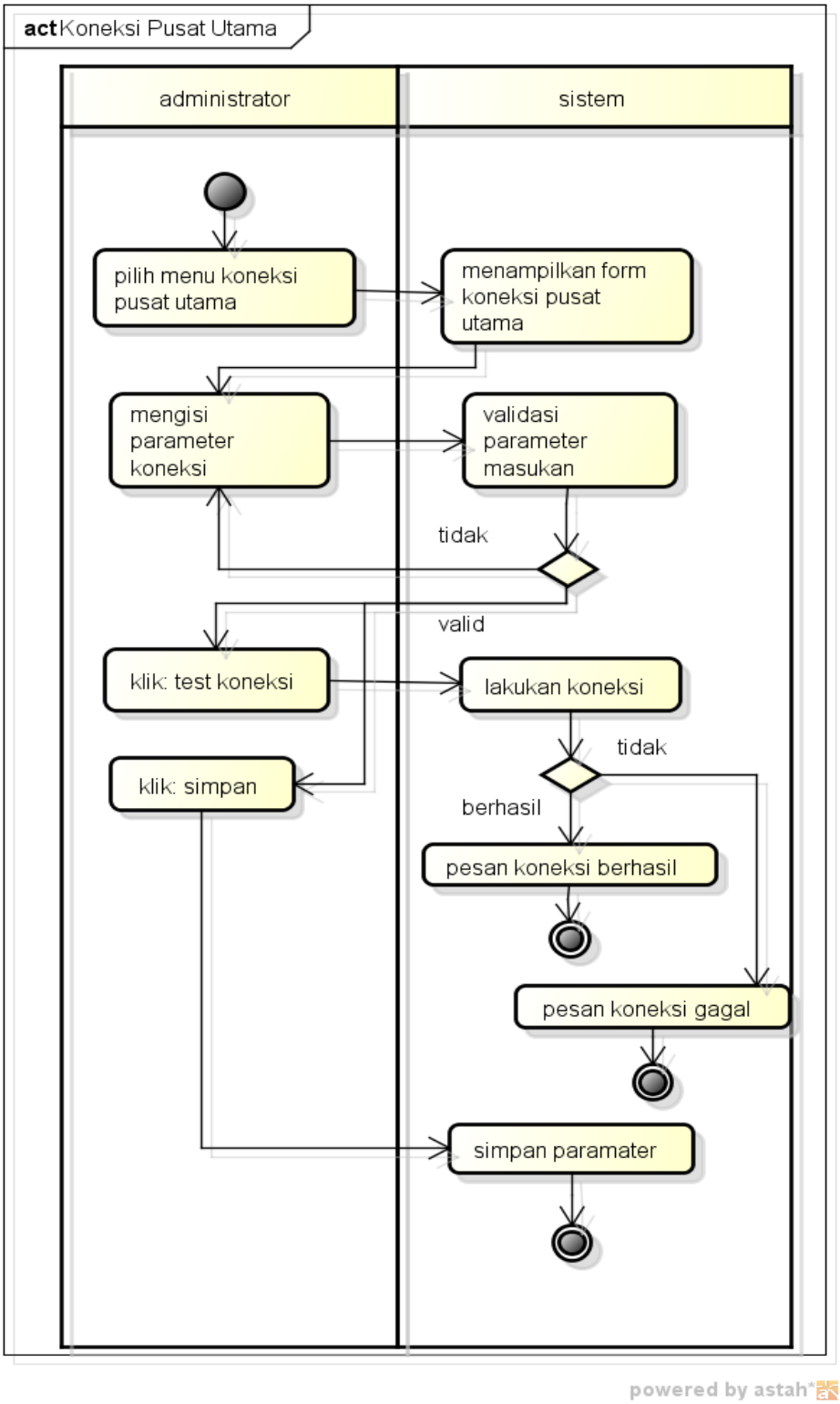
Gambar 3. 33 Activity Diagaram Ubah Password

3.1.6.5 Skenario *Use Case* Pengaturan Koneksi Pusat Utama

Proses ini akan melakukan operasi untuk mengnyambungkan antara sistem dan *database* utama pusat.

Tabel 3. 10 Skenario *Use Case* Pengaturan Koneksi Pusat Utama

Identifikasi	
Nomor	3
Nama	Pengaturan Koneksi Pusat Utama
Tujuan	Melakukan koneksi dari sistem ke *Database* pusat utama
Deskripsi	Administrator melakukan koneksi dari sistem ke *Database* pusat utama
Aktor	Administrator
Skenario Utama	
Kondisi awal	Administrator memiliki hak akses. Menampilkan form pengaturan koneksi.
Aksi Aktor	**Reaksi Sistem**
1. Administrator memilih menu pengaturan koneksi	2. Menampilkan form pengaturan koneksi *database* utana pusat
3. Administrator memasukan host, user, password, dan *database*.	4. Sistem memvalidasi parameter masukan
5. Klik tombol tes koneksi	6. Sistem akan melakukan proses koneksi ke *Database* tujuan, dan memberikan respon terhdap status koneksi.
7. Klik tombol simpan	8. Sistem akan menyimpan parameter koneksi
Kondisi akhir	Koneksi *Database* pusat utama terjadi.



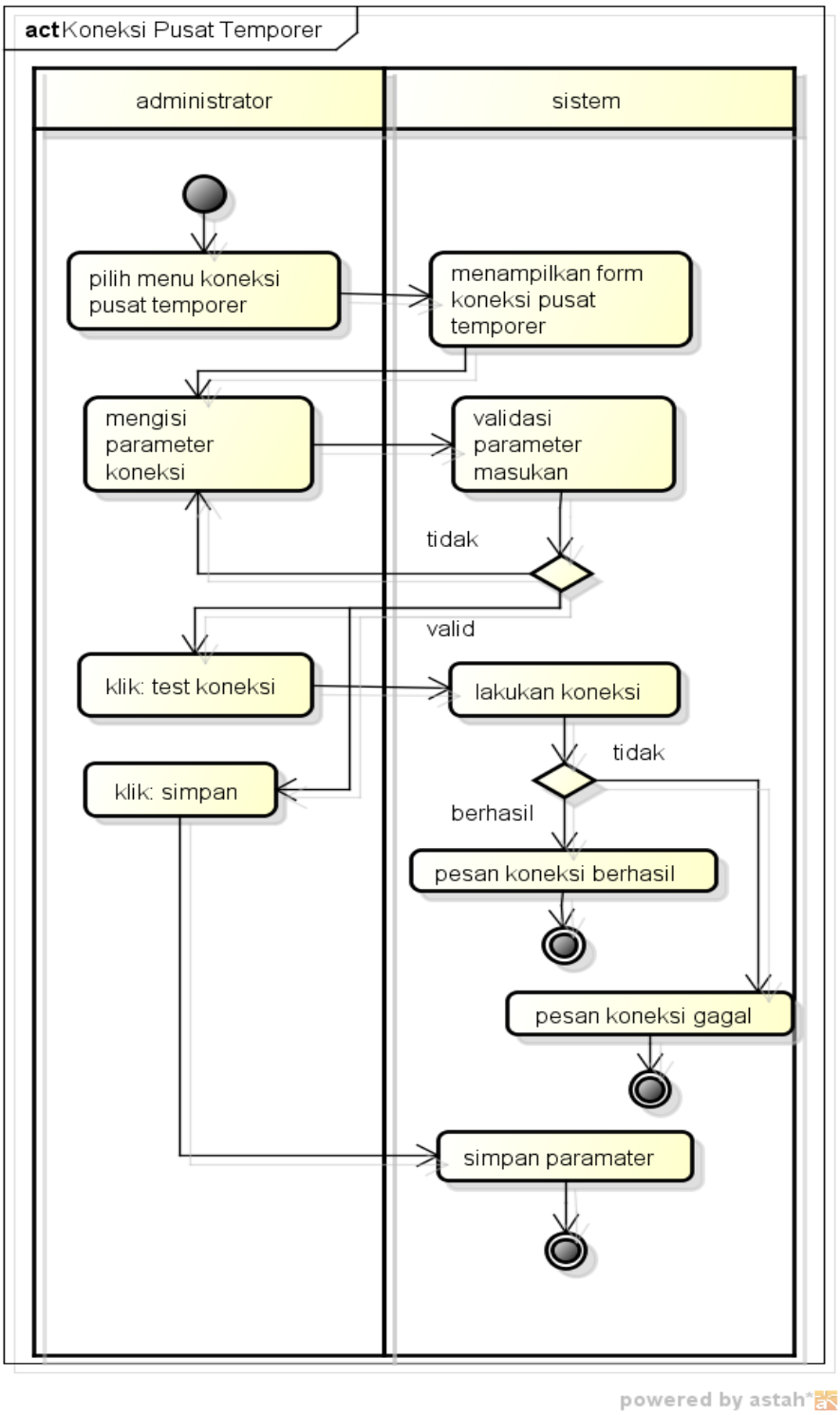
Gambar 3. 34 Activity Diagram Pengaturan Koneksi *Database* Pusat Utama

3.1.6.6 Skenario *Use Case* Pengaturan Koneksi Pusat Temporer

Proses ini akan melakukan operasi untuk mengnyambungkan antara sistem dan *database* pusat temporer.

Tabel 3. 11 Skenario *Use Case* Pengaturan Koneksi Pusat Temporer

Identifikasi	
Nomor	4
Nama	Pengaturan Koneksi Pusat Temporer
Tujuan	Melakukan koneksi dari sistem ke *Database* pusat temporer
Deskripsi	Administrator melakukan koneksi dari sistem ke *Database* pusat temporer
Aktor	Administrator
Skenario Utama	
Kondisi awal	Administrator memiliki hak akses. Menampilkan form form administrator
Aksi Aktor	**Reaksi Sistem**
1. Administrator memilih menu pengaturan koneksi	2. Menampilkan form pengaturan koneksi *database* temp pusat
3. Administrator memasukan host, user, password, dan *database*.	4. Sistem memvalidasi parameter masukan
5. Klik tombol tes koneksi	6. Sistem akan melakukan proses koneksi ke *Database* tujuan, dan memberikan respon terhdap status koneksi.
7. Klik tombol simpan	8. Sistem akan menyimpan parameter koneksi
Kondisi akhir	Koneksi *Database* pusat temporer terjadi.



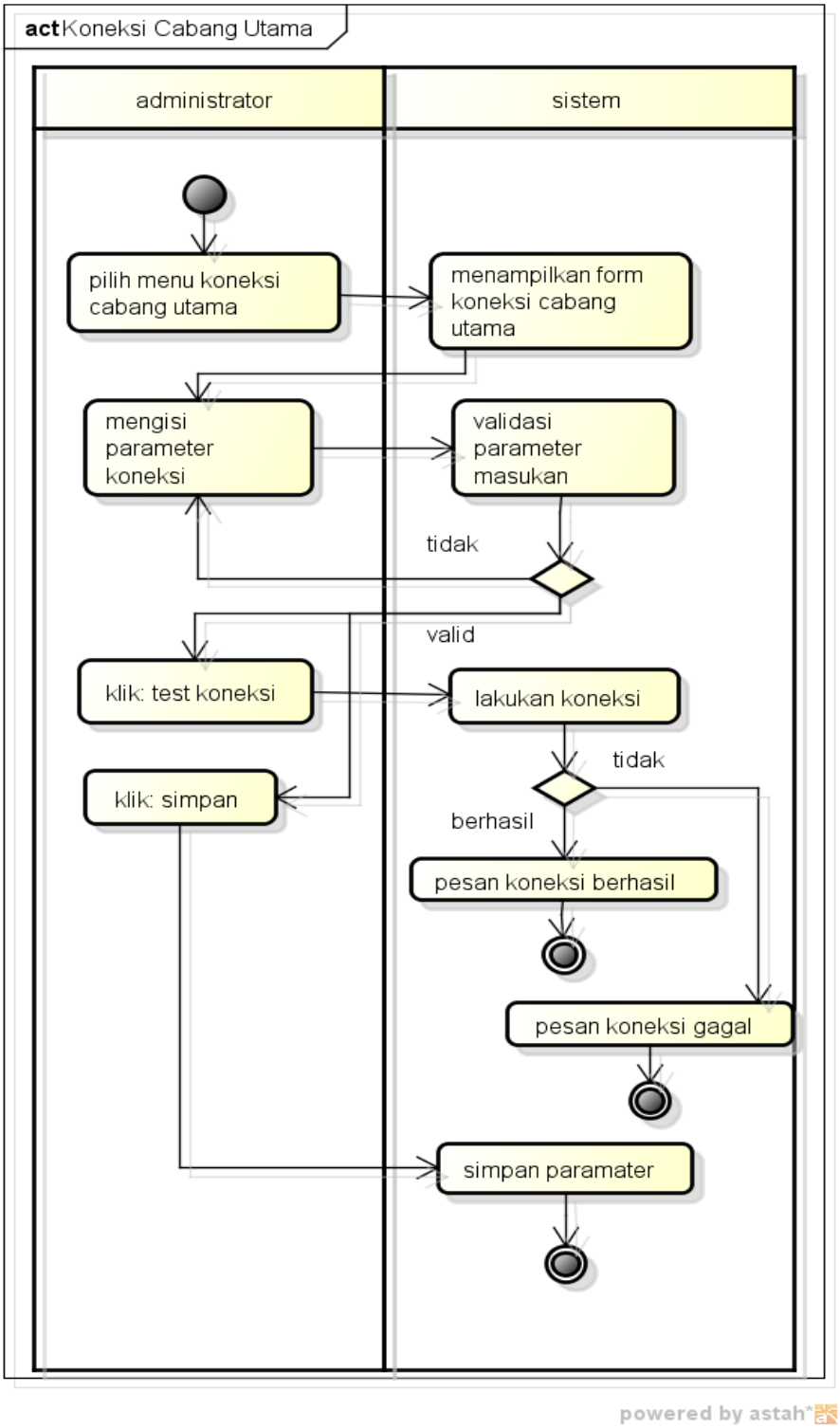
Gambar 3. 35 Activity Diagram Pengaturan Koneksi *Database*Pusat Utama

3.1.6.7 Skenario Use Case Pengaturan Koneksi Cabang Utama

Proses ini akan melakukan operasi untuk mengnyambungkan antara sistem dan *database* cabang temporer.

Tabel 3. 12 Skenario *Use Case* Pengaturan Koneksi Cabang Utama

Identifikasi		
Nomor	5	
Nama	Pengaturan Koneksi Cabang Utama	
Tujuan	Melakukan koneksi dari sistem ke *Database* Cabang utama	
Deskripsi	Administrator melakukan koneksi dari sistem ke *Database* cabang utama	
Aktor	Administrator	
Skenario Utama		
Kondisi awal	Administrator memiliki hak akses. Menampilkan form pengaturan koneksi.	
Aksi Aktor		**Reaksi Sistem**
1. Administrator memilih menu pengaturan koneksi		2. Menampilkan form pengaturan koneksi *database* utama cabang
3. Administrator memasukan host, user, password, dan *database*.		4. Sistem memvalidasi parameter masukan
5. Klik tombol tes koneksi		6. Sistem akan melakukan proses koneksi ke *Database* tujuan, dan memberikan respon terhdap status koneksi.
7. Klik tombol simpan		8. Sistem akan menyimpan parameter koneksi
Kondisi akhir	Koneksi *Database* cabang utama terjadi.	



Gambar 3. 36 Activity Diagram Pengaturan Koneksi *Database* Utama Cabang

3.1.6.8 Skenario *Use Case* Pengaturan Koneksi CabangTemporer

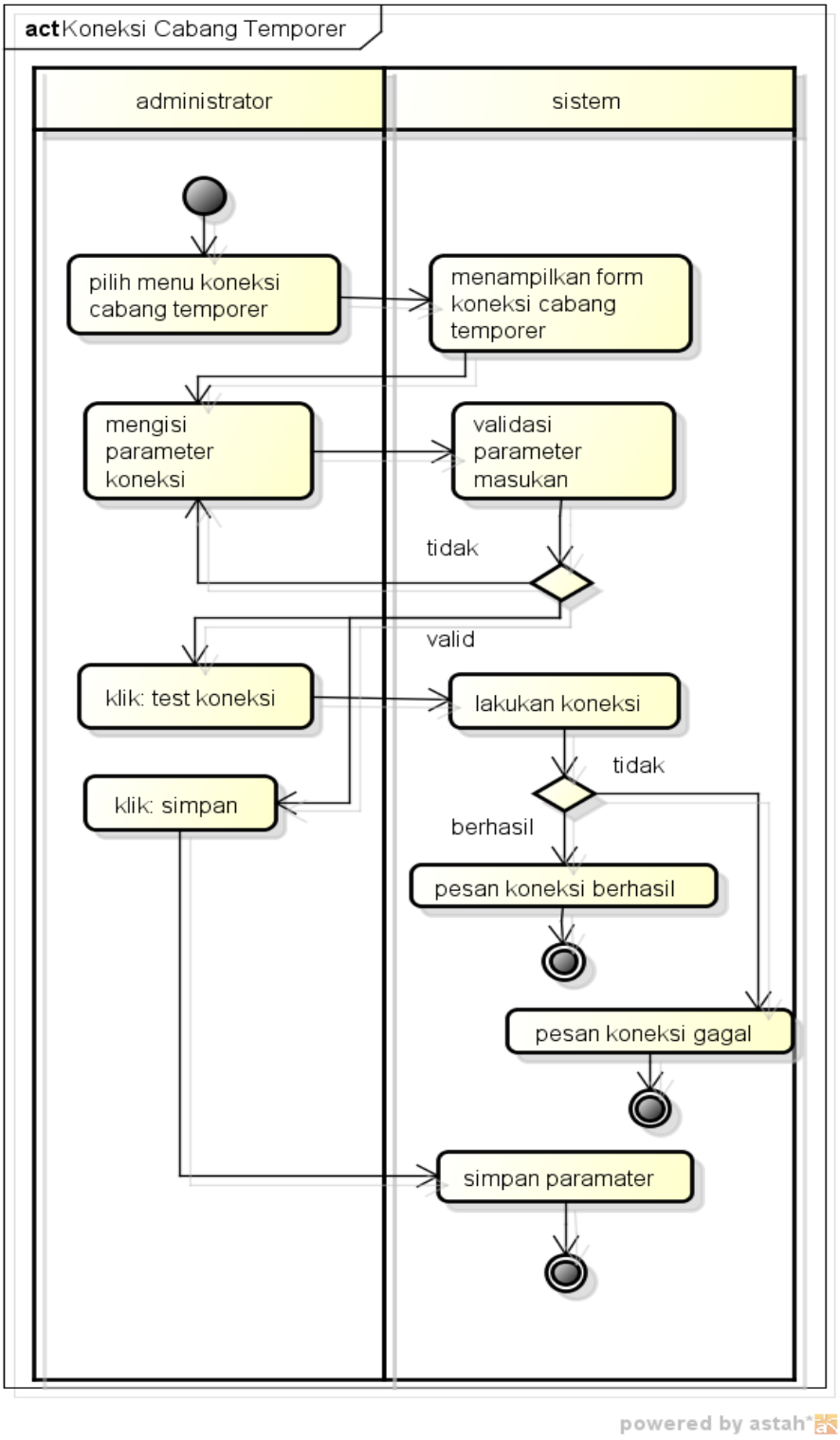
Proses ini akan melakukan operasi untuk mengnyambungkan antara sistem dan *database* cabang temporer.

Tabel 3. 13 Skenario *Use Case* Pengaturan Koneksi Cabang Temporer

Identifikasi	
Nomor	6
Nama	Pengaturan Koneksi Cabang Temporer
Tujuan	Melakukan koneksi dari sistem ke *Database* pusat temporer
Deskripsi	Administrator melakukan koneksi dari sistem ke *Database* cabang temporer
Aktor	Administrator
Skenario Utama	
Kondisi awal	Administrator memiliki hak akses. Menampilkan form pengaturan koneksi.

Aksi Aktor	**Reaksi Sistem**
1. Administrator memilih menu pengaturan koneksi	2. Menampilkan form pengaturan koneksi *database* utama cabang
3. Administrator memasukan host, user, password, dan *database*.	4. Sistem memvalidasi parameter masukan
5. Klik tombol tes koneksi	6. Sistem akan melakukan proses koneksi ke *Database* tujuan, dan memberikan respon terhdap status koneksi.
7. Klik tombol simpan	8. Sistem akan menyimpan parameter koneksi

Kondisi akhir	Koneksi *Database* pusat temporer terjadi.



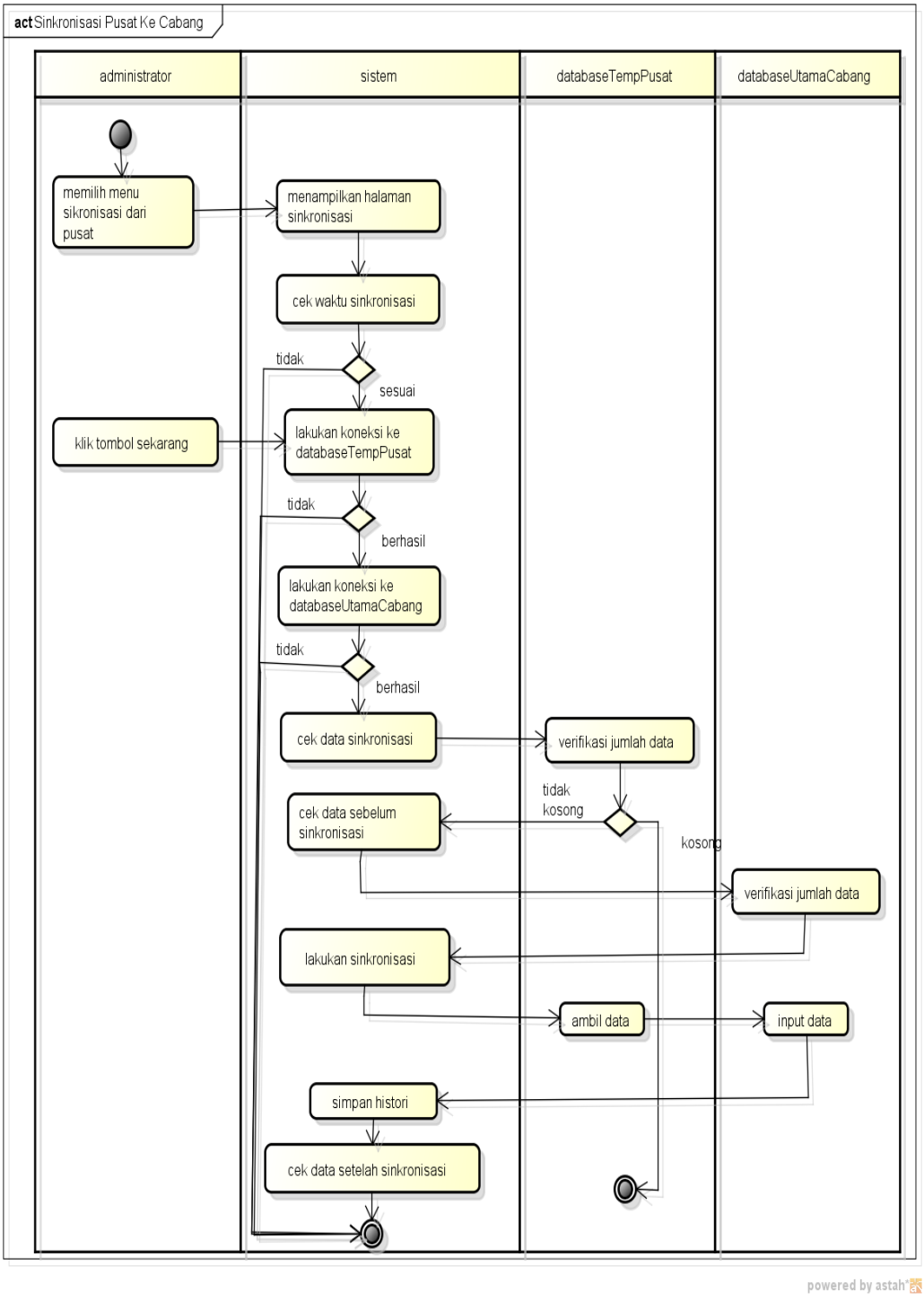
Gambar 3. 37 Activity Diagram Pengaturan Koneksi *Database* Utama Cabang

3.1.6.9 Skenario *Use Case* Sinkronisasi *Database* Pusat Ke Cabang

Proses ini menjelaskan proses proses sinkronisasi dari *databas* pusat ke *database* cabang

Tabel 3. 14 Skrenario Use Case Sinkronisasi *Database* Pusat Ke Cabang

Identifikasi	
Nomor	7
Nama	Sinkronisasi *Database* Pusat Ke Cabang
Tujuan	Melakukan sinkronisasi *Database* dari *Database* pusat ek *Database* cabang agar *Database* berjalan *uptodate*
Deskripsi	Sinkronisasi *Database* pusat ke *Database* cabang dilakukan dengan memeriksa *Database* temporer pusat kemudian membandingkan datanya dengan *Database* utama cabang. Jika ada yang perlu disinkronisasikan maka sinkronisasi akan dijalankan.
Aktor	Administrator, *database*TempPusat, *database*UtamaPusat
Skenario Utama	
Kondisi awal	Administrator memiliki hak akses , koneksi *Database* berjalan dengan baik dan menampilkan form menu sinkronisasi *Database* puast ke cabang
Aksi Aktor	**Reaksi Sistem**
1. Administrator memilih menu sinkronisasi.	2. Sistem akan merespon menu yang dipilih dengan menampilkan form yang sesuai.
3. Administrator memasukan parameter yang diperlukan.	4. Sistem akan melakukan proses sinkronisasi sesuai parameter yang diberikan.
Kondisi akhir	Sinkronisai *database* dilakukan.

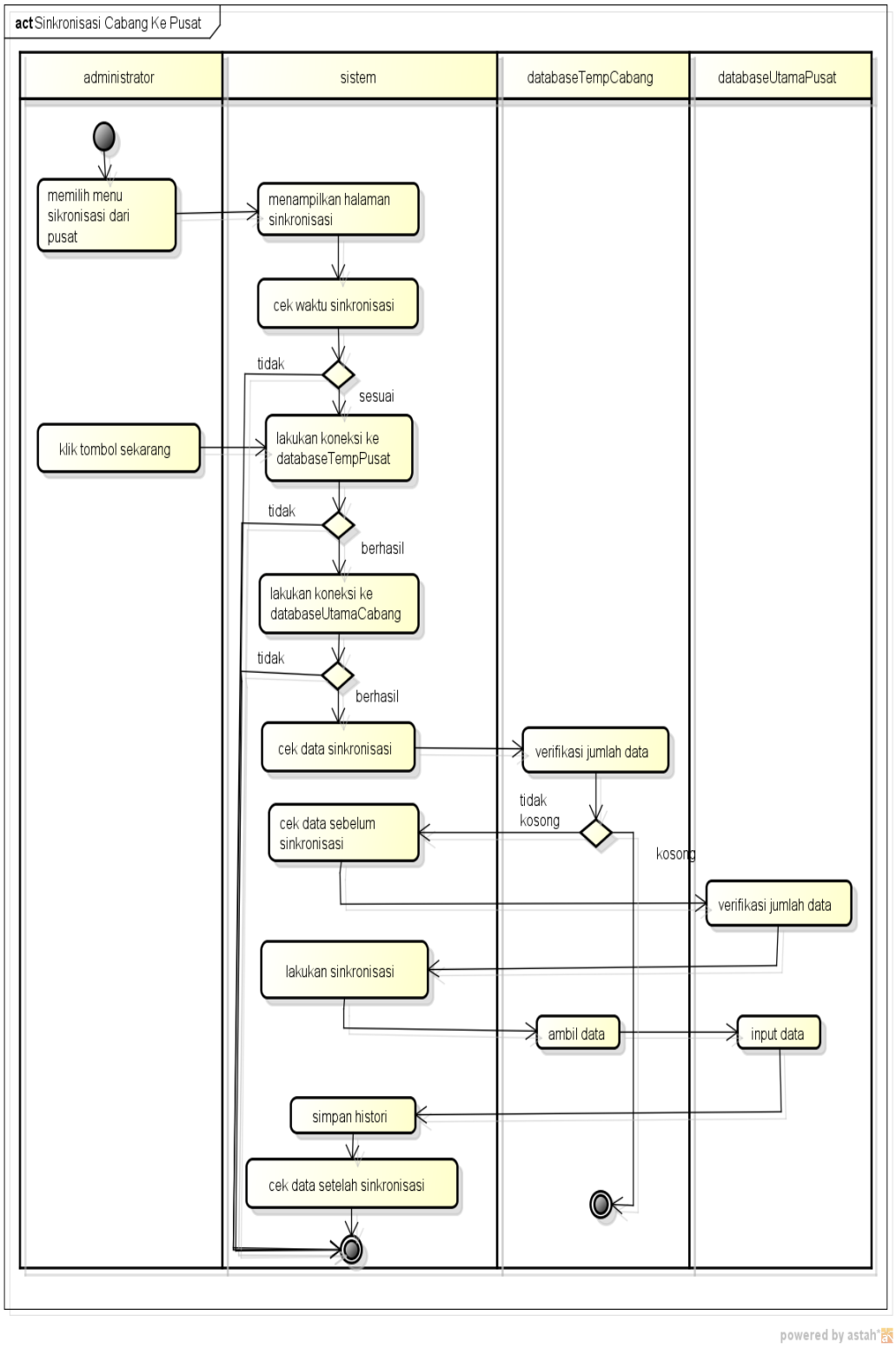


Gambar 3. 38 Activity Diagram Sinkronisasi *Database* Pusat Ke Cabang

3.1.6.10 Skenario *Use Case* Sinkronisasi *Database* Cabang Ke Pusat

Tabel 3. 15 Skrenario *Use Case* Sinkronisasi *Database* Cabang Ke Pusat

Identifikasi	
Nomor	8
Nama	Sinkronisasi *Database* cabang ke pusat
Tujuan	Melakukan sinkronisasi *Database* dari *Database* cabang ke *Database* pusat agar *Database* berjalan *uptodate*
Deskripsi	Sinkronisasi *Database* cabang ke *Database* pusat dilakukan dengan memeriksa *Database* temporer cabang kemudian membandingkan datanya dengan *Database* utama pusat. Jika ada yang perlu disinkronisasikan maka sinkronisasi akan dijalankan.
Aktor	Administrator, *database*TenpCabang, *Database*UtamaPusat
Skenario Utama	
Kondisi awal	Administrator memiliki hak akses , koneksi *Database* berjalan dengan baik dan menampilkan form menu sinkronisasi *Database* c
Aksi Aktor	**Reaksi Sistem**
1. Administrator memilih menu sinkronisasi.	2. Sistem akan menampilkan form sinkronisasi.
3. Administrator memasukan parameter yang diperlukan.	4. Sistem akan melakukan proses sinkronisasi sesuai parameter yang diberikan.
Kondisi akhir	Sistem melaksanakan proses sinkronisasi *Database.*

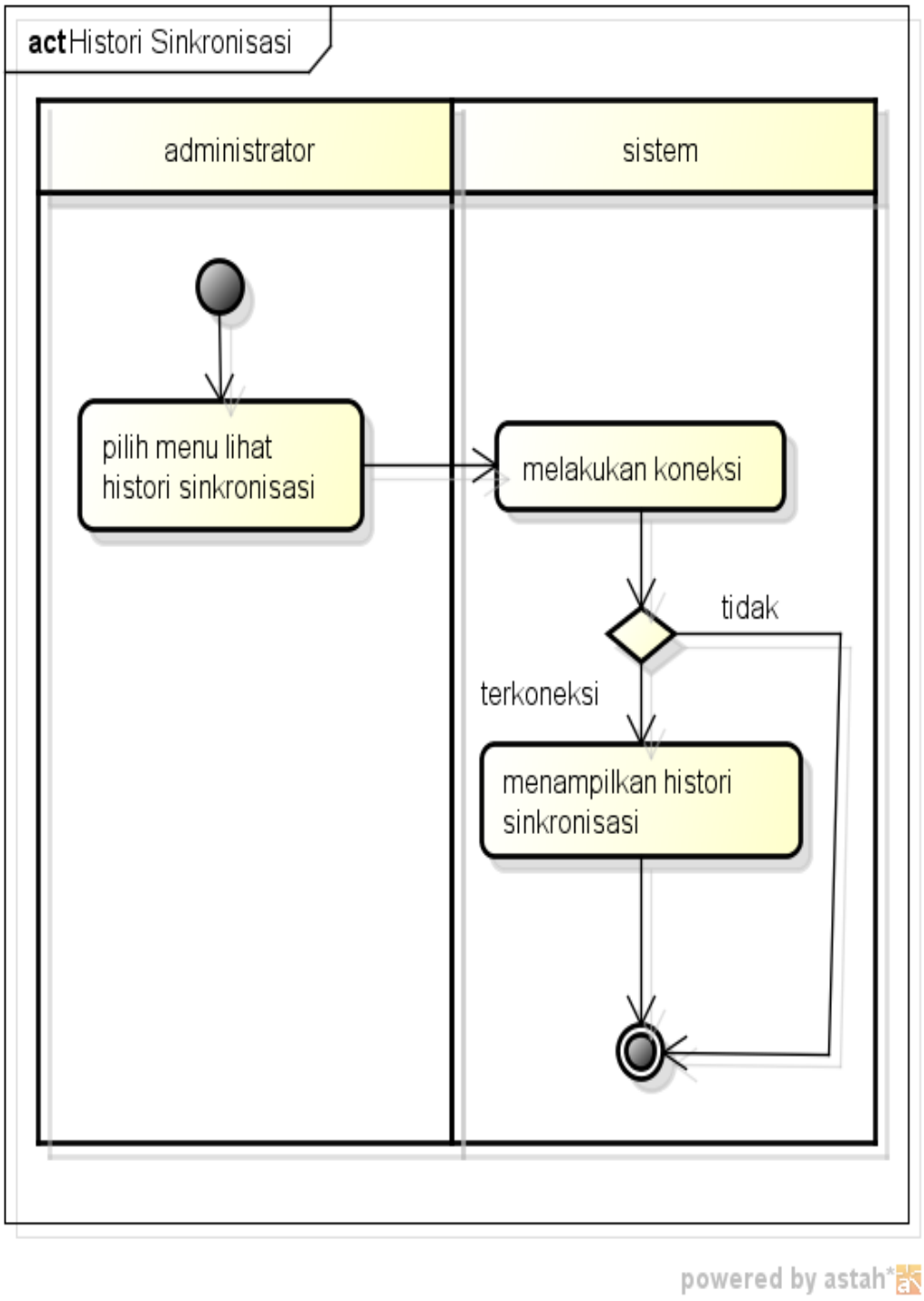


Gambar 3. 39 Activity Diagram Sinkronisasi *Database* Cabang Ke Pusat

3.1.6.11 Skenario *Use Case* Lihat Histori Sinkronisasi

Tabel 3. 16 Skrenario *Use Case* Lihat Histori Sinkronisasi

Identifikasi		
Nomor	9	
Nama	Lihat Histori Sinkronisasi	
Tujuan	Melihat histori atau riwayat sinkroniasi yang telah dilakukan sebelumnya	
Deskripsi	Sinkronisasi yang dilakukan akan dicatat, baik berhasil atau pun gagal. Use case ini bertujuan untuk melihat daftar sinkroniassi yang telah dilakukan	
Aktor	Administrator	
Skenario Utama		
Kondisi awal	Sinkronisasi telah dilakukan	
Aksi Aktor		**Reaksi Sistem**
1. Administrator memilih menu lihat sinkroniasi		2. Sistem akan merespon menu yang dipilih dengan menampilkan form yang sesuai.
Kondisi akhir	Sistem akan menampilkan riawayat sinkronisasi	

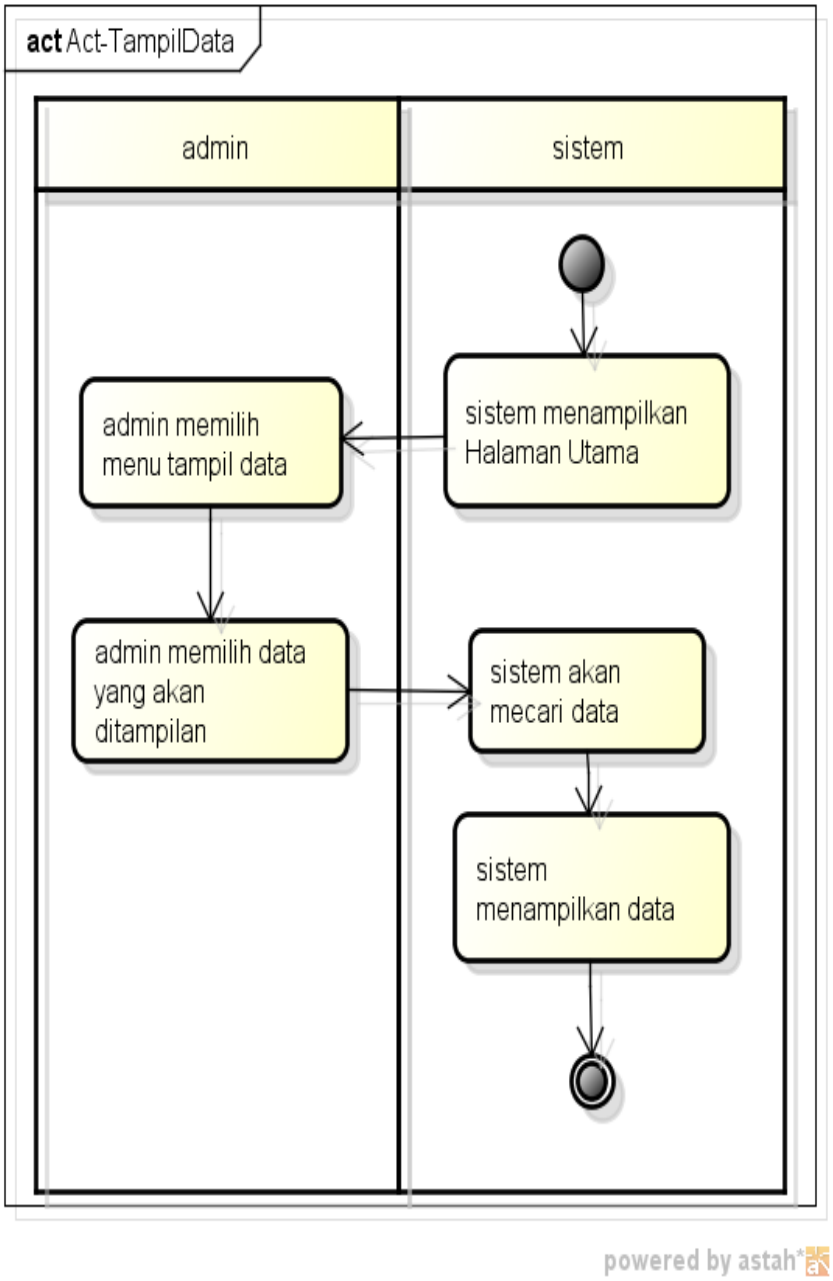


Gambar 3. 40 Activity Diagram Lihat Histori Sinkronisasi

3.1.6.12 Skenario *Use Case* Tampil *Data*

Tabel 3. 17 Skenario Use Case Tampil Data

Identifikasi		
Nomor	10	
Nama	Tampil data	
Tujuan	Menampilkan data	
Deskripsi	Sistem akan menampilkan data yang ada dalam *Database* pusat	
Aktor	Administrator	
Skenario Utama		
Kondisi awal	*Database* pusat tidak kosong	
Aksi Aktor		**Reaksi Sistem**
Aktor akan memilih tabel mana yang akan ditampilkan datanya		Sistem akan menampilkan data sesuai dengan permintaan admin
Kondisi akhir	Data ditampilkan	



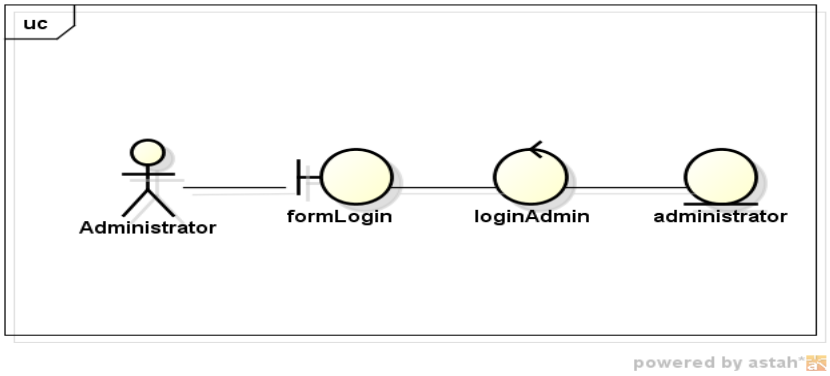
Gambar 3. 41 Activity Diagram Tampil Data

3.1.7 *Model* Analisis *Use Case*

3.1.7.1 *Use* Case Login

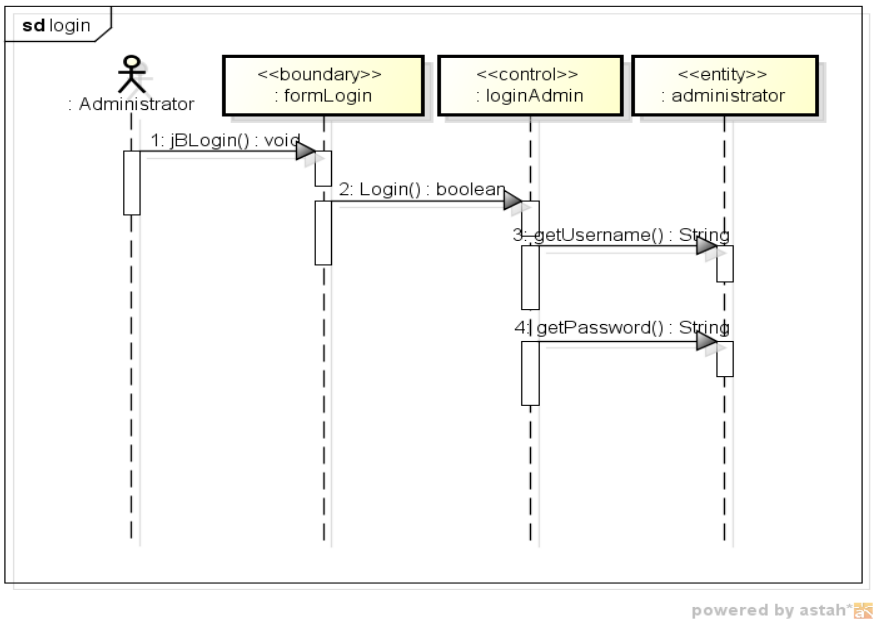
Hubungan interaksi antar objek *use case* login admin digambarkan sebagia berikut :

A. Kelas Analisis Login



Gambar 3. 42 Kelas Analisi Login

B. *Sequence diagram* Login

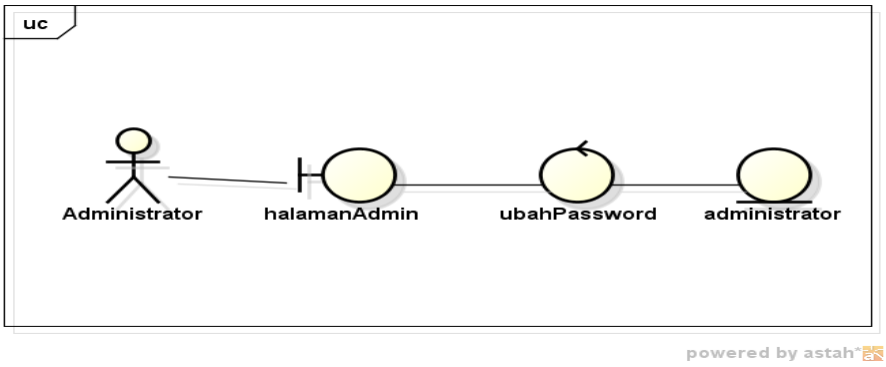


Gambar 3. 43 Kelas *Sequence diagram* Login

3.1.7.2 *Use Case* Ubah *Password*

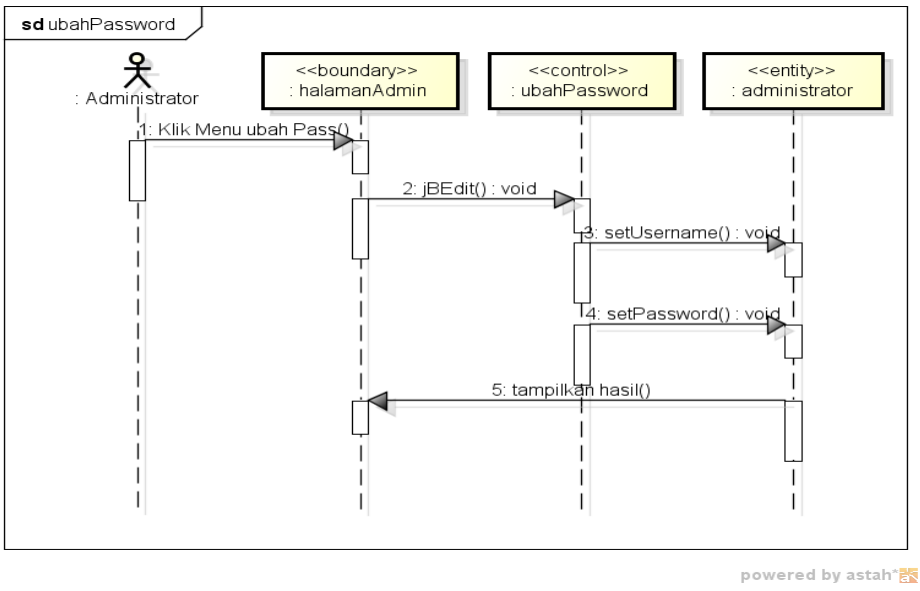
Hubungan antar objek ubah password digambarkan sebagai berikut:

A. Kelas Analisis Ubah *Password*



Gambar 3. 44 Kelas Analisis Ubah Password

B. *Sequence diagram* Ubah *Password*

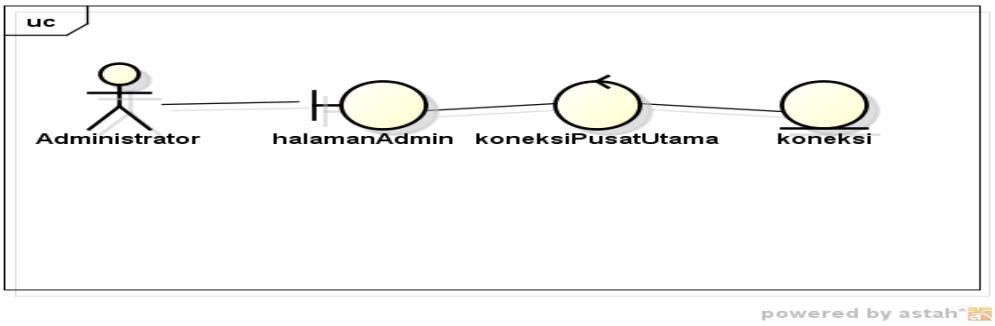


Gambar 3. 45*Sequence diagram* Ubah Password

3.1.7.3 *Use Case* Pengaturan Koneksi Pusat Utama

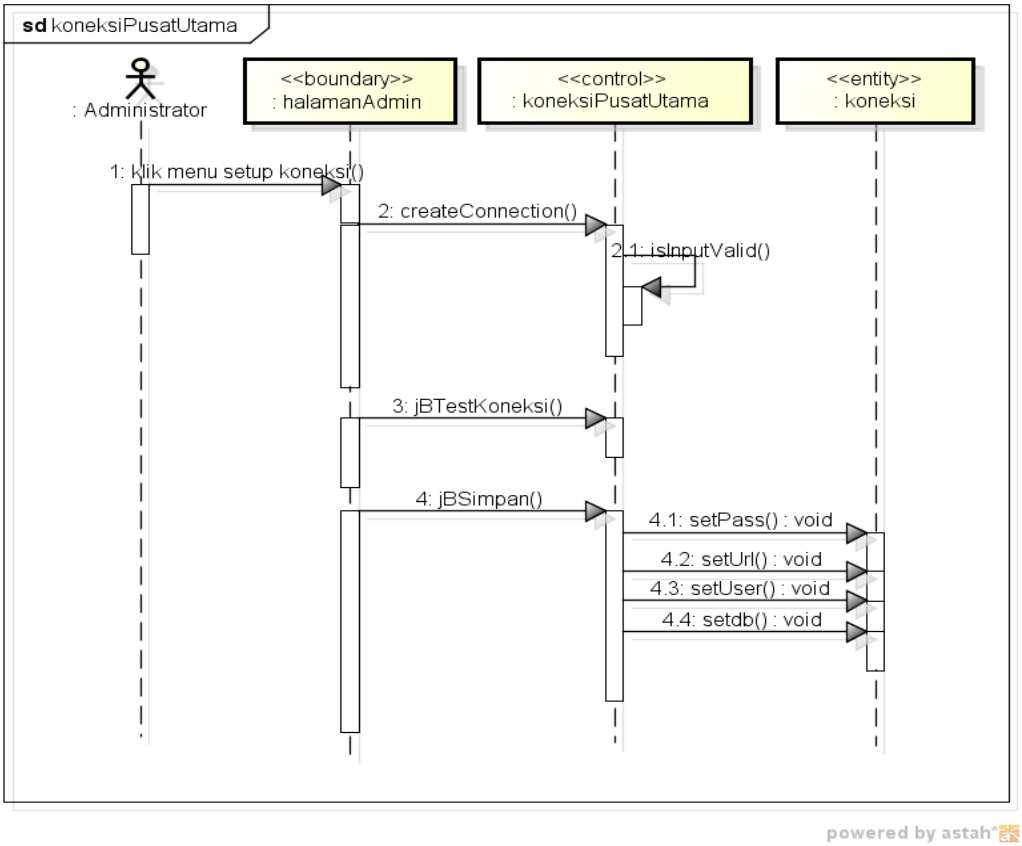
Hubungan antar objek pengaturan koneksi digambarkan sebagai berikut:

A. Kelas Analisis Pengaturan Koneksi Pusat Utama



Gambar 3. 46 Pengaturan Koneksi Pusat Utama

B. *Sequence diagram* Pengaturan Koneksi Pusat Utama

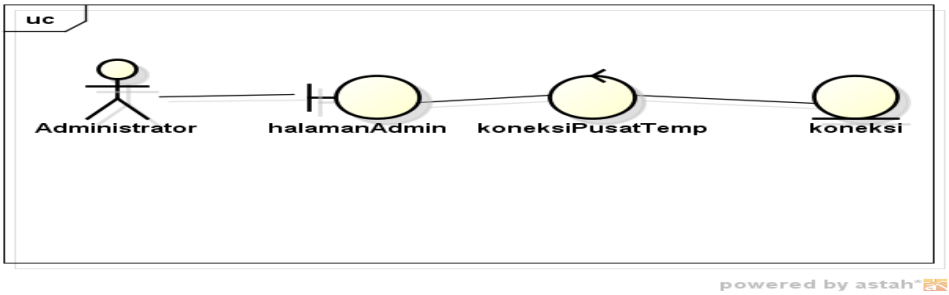


Gambar 3. 47 *Sequence diagram* Pengaturan Koneksi Pusat Utama

3.1.7.4 *Use Case* Pengaturan Koneksi Pusat Temporer

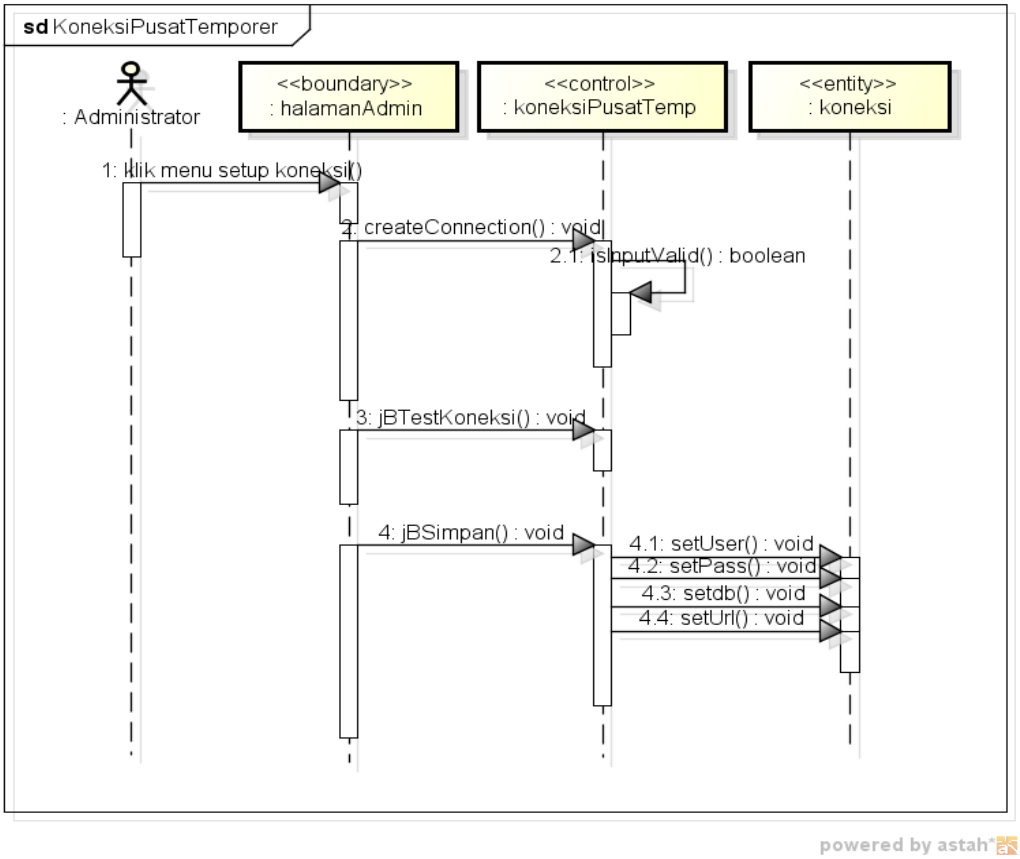
Hubungan antar objek pengaturan koneksi digambarkan sebagai berikut:

A. Kelas Analisis Pengaturan Koneksi Temporer



Gambar 3. 48 Kelas Analisis Pengaturan Koneksi Pusat Temporer

B. *Sequence diagram* Pengaturan Koneksi Pusat Temporer

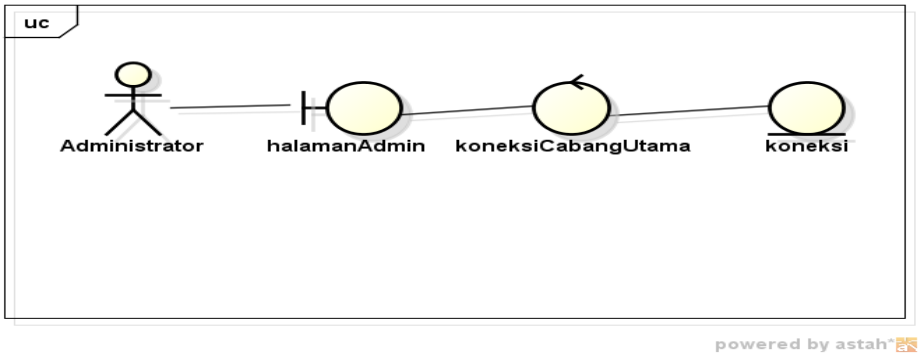


Gambar 3. 49 *Sequence diagram* Pengaturan Koneksi Cabang Temporer

3.1.7.5 *Use Case* Pengaturan Koneksi Cabang Utama

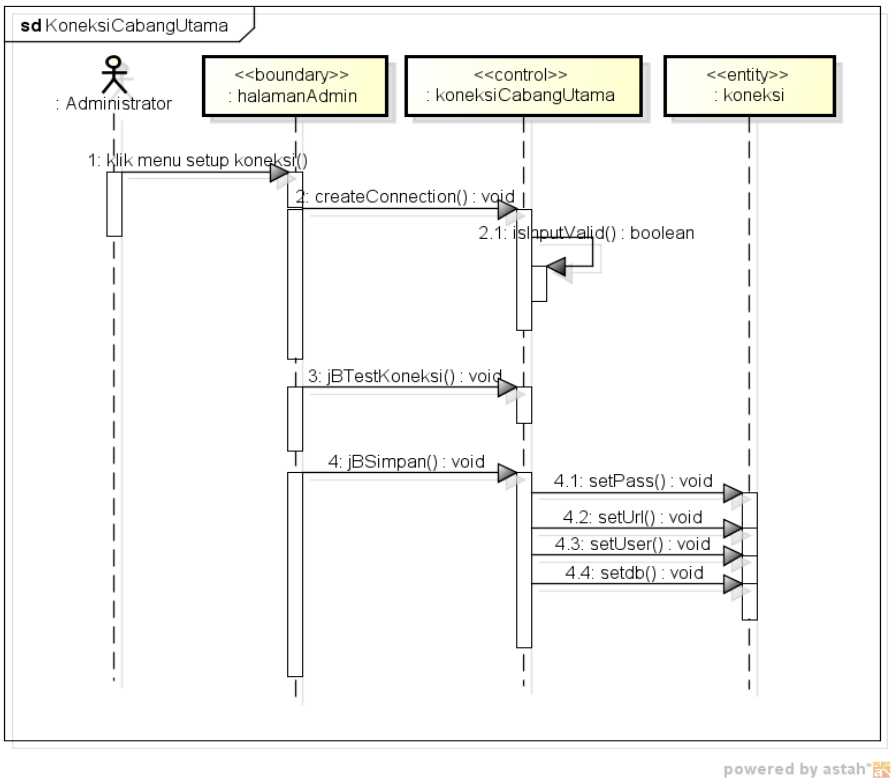
Hubungan antar objek pengaturan koneksi digambarkan sebagai berikut:

A. Kelas Analisis Pengaturan Koneksi Cabang Utama



Gambar 3. 50 Kelas Analisis Pengaturan Koneksi Cabang Utama

B. *Sequence diagram* Pengaturan Koneksi Cabang Utama

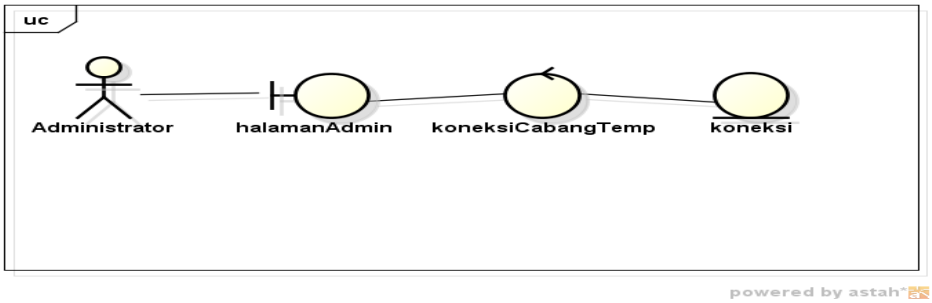


Gambar 3. 30 *Sequence diagram* Pengaturan Koneksi Cabang Utama

3.1.7.6 *Use Case* Pengaturan Koneksi Cabang Temporer

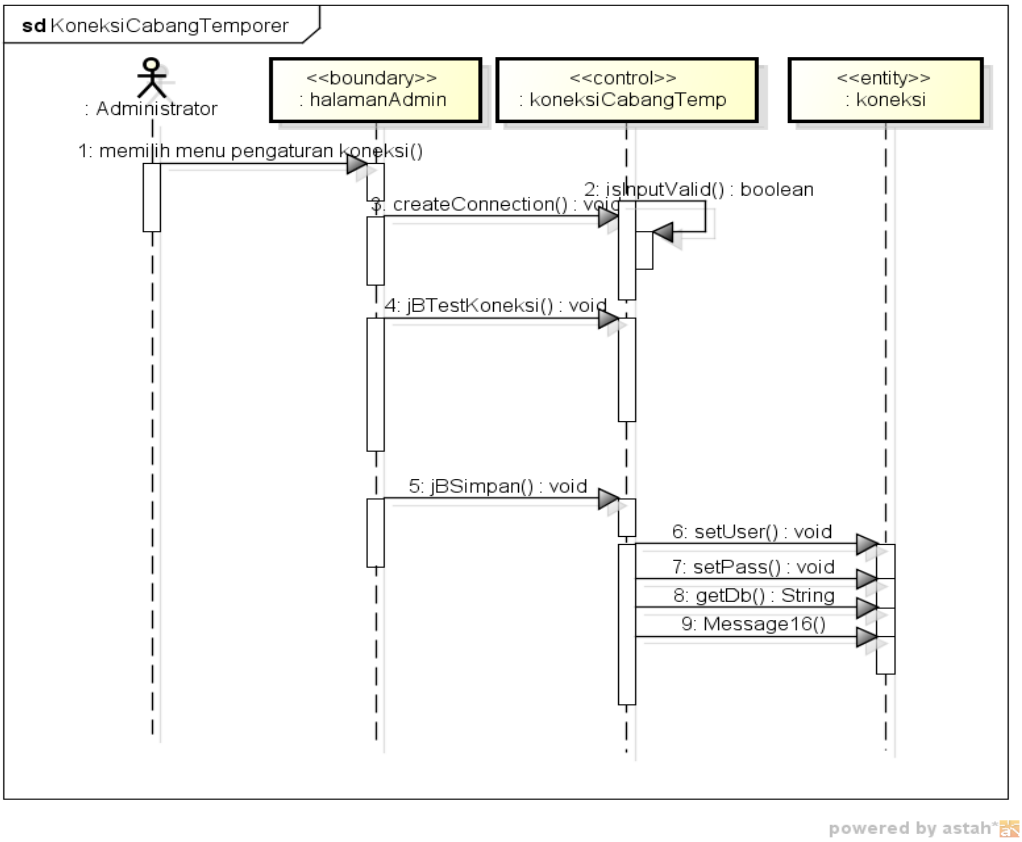
Hubungan antar objek pengaturan koneksi digambarkan sebagai berikut:

A. Kelas Analisis Pengaturan Koneksi Cabang Temporer



Gambar 3. 51 Kelas Analisis Koneksi Cabang Temporer

B. *Sequence diagram* Pengaturan Koneksi Cabang Temporer

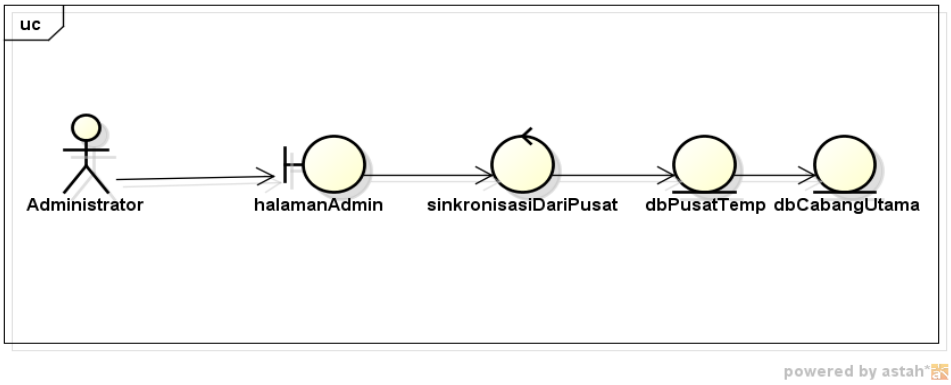


Gambar 3. 52 *Sequence diagram* Pengaturan Koneksi Cabang Temporer

3.1.7.7 *Use Case Sinkoronisasi Database* Pusat Ke Cabang

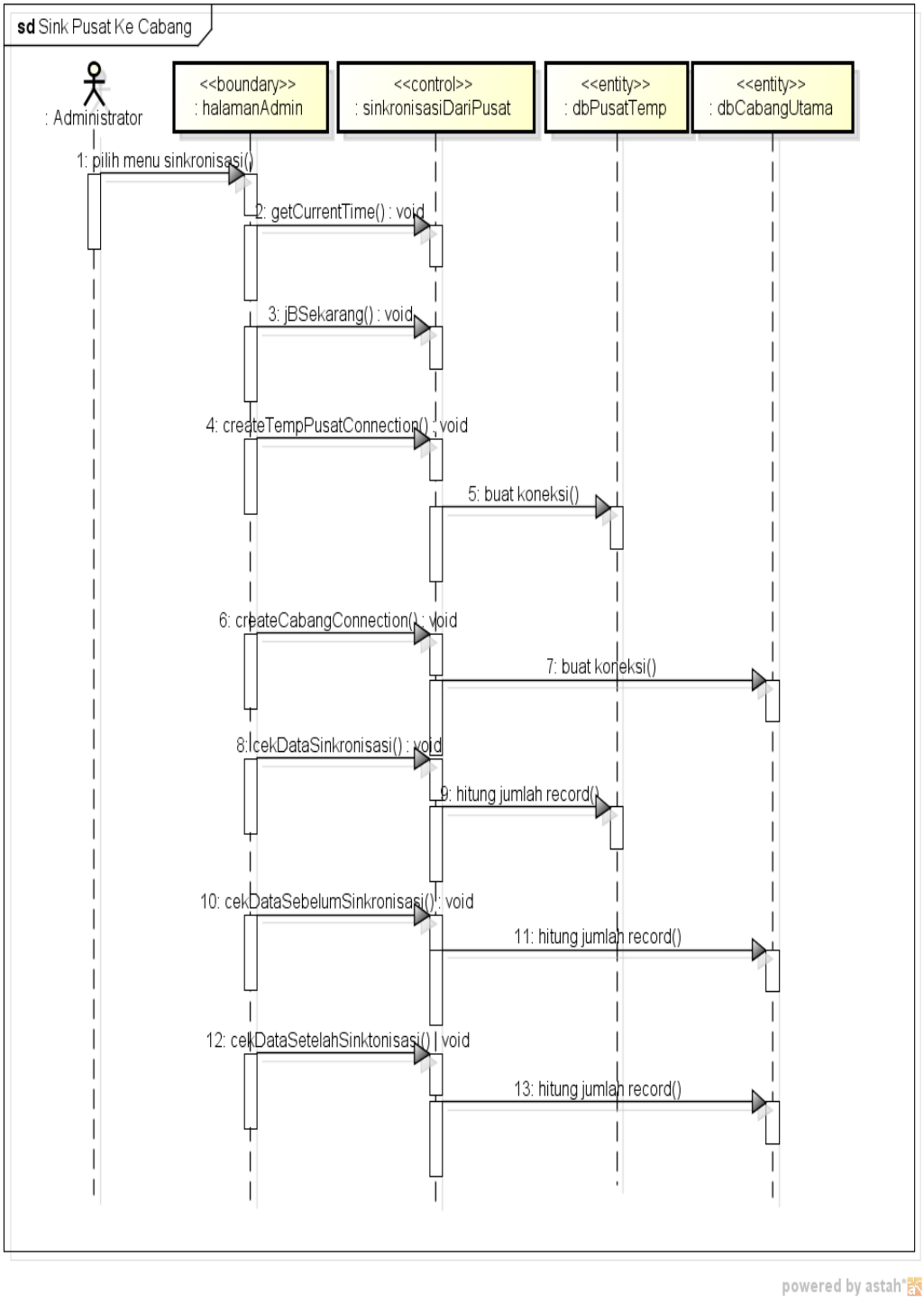
Hubungan antar objek *sinkronisasi Database* digambarkan sebagai berikut:

A. Kelas Analisis *Sinkronisasi Database*



Gambar 3. 539 Kelas Analisis Sinkronisasi *Database* Pusat Ke Cabang

B. *Sequence diagram* Sinkronisasi *Database* Pusat Ke Cabang

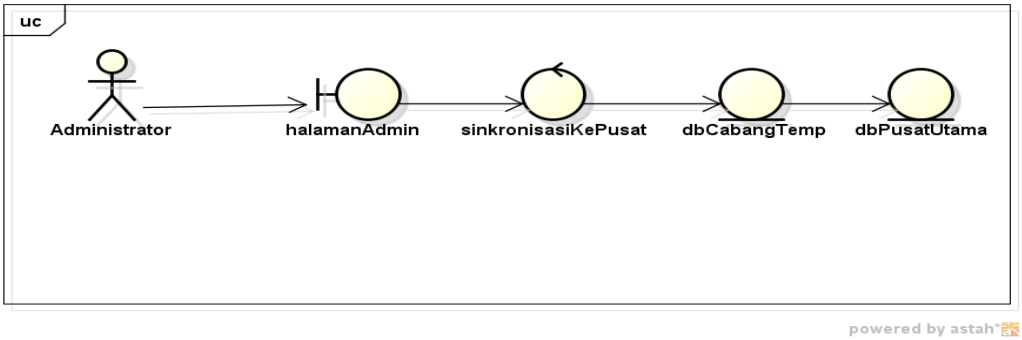


Gambar 3. 30 *Sequence diagram* Sinkronisasi *Database* Pusat Ke Cabang

3.1.7.8 *Use Case* Sinkronisasi *Database* Cabang Ke Pusat

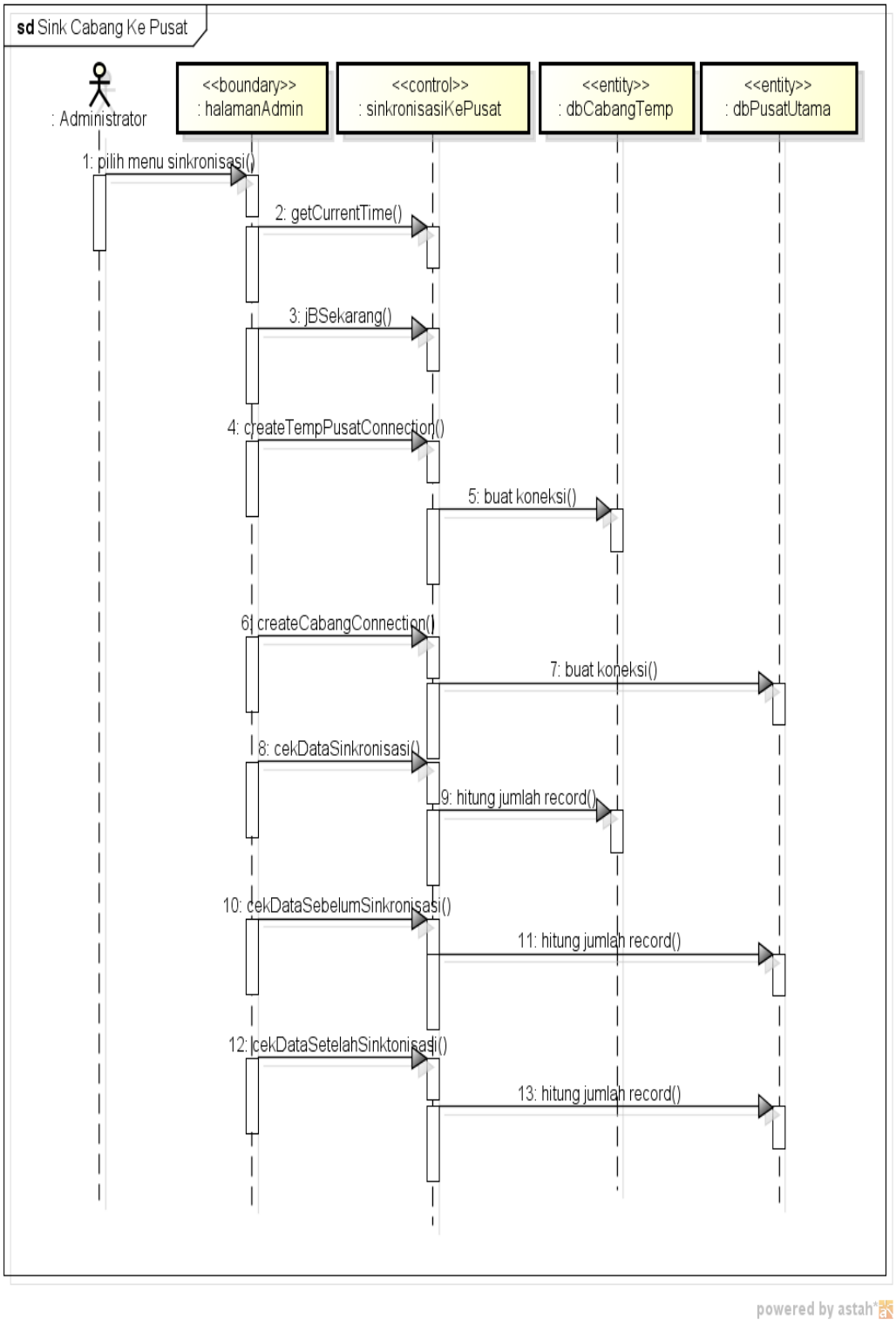
Hubungan antar objek sinkronisasi *Database* digambarkan sebagai berikut:

A. Kelas Analisis Sinkronisasi *Database* Cabang Ke Pusat



Gambar 3. 32 Kelas Analisis Sinkronisasi *Database* Cabang Ke Pusat

B. *Sequence diagram* Sinkronisasi *Database* Cabang Ke Pusat

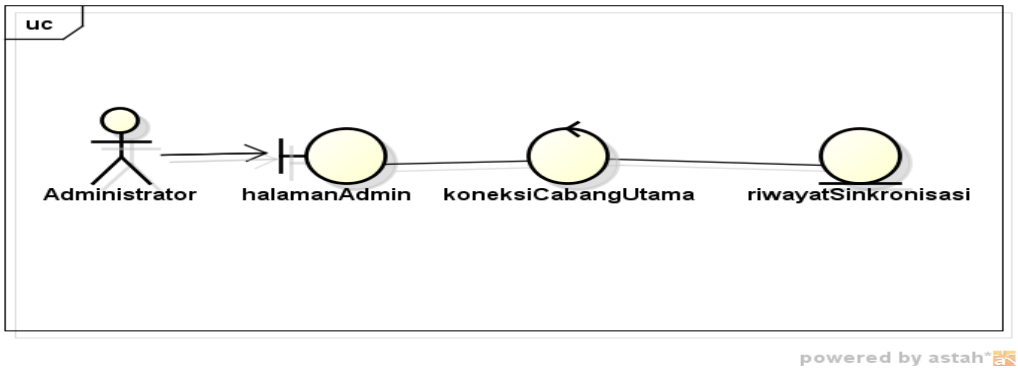


Gambar 3. 54 *Sequence diagram* Sinkronisasi *Database* Cabang Ke Pusat

3.1.7.9 *Use Case* Lihat Histori Sinkronisasi

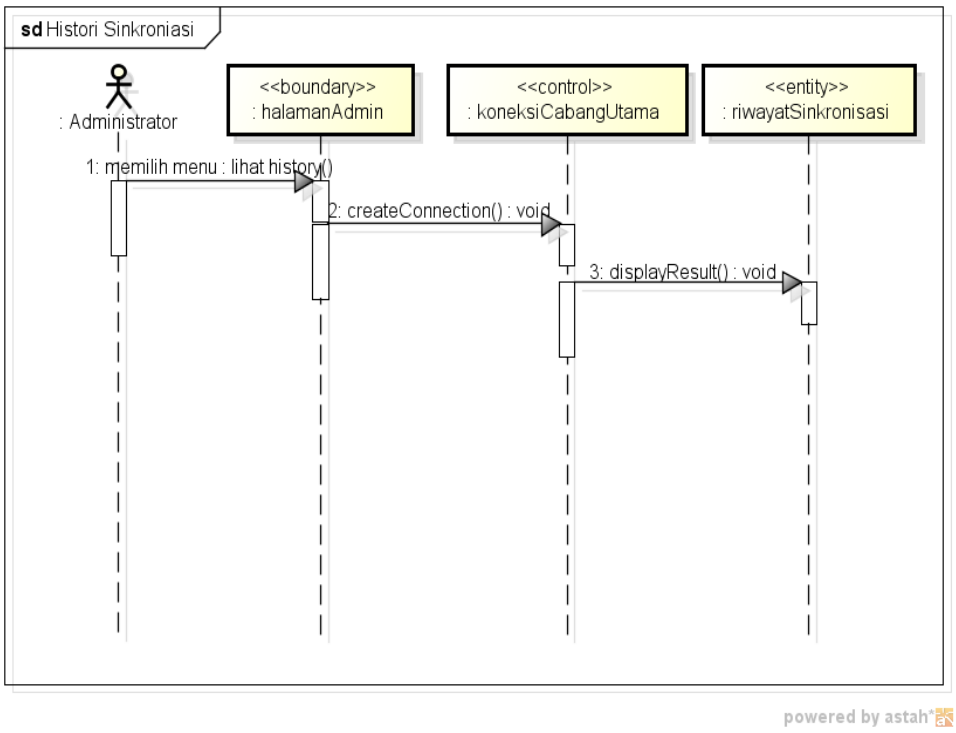
Hubungan antar histori sinkroniasasi digambarkan sebagai berikut:

C. Kelas Analisis Lihat History Sinkroniasi



Gambar 3. 32 Kelas Analisis Lihat Histori Sinkronisasi

D. *Sequence diagram* Lihat History Sinkroniasi

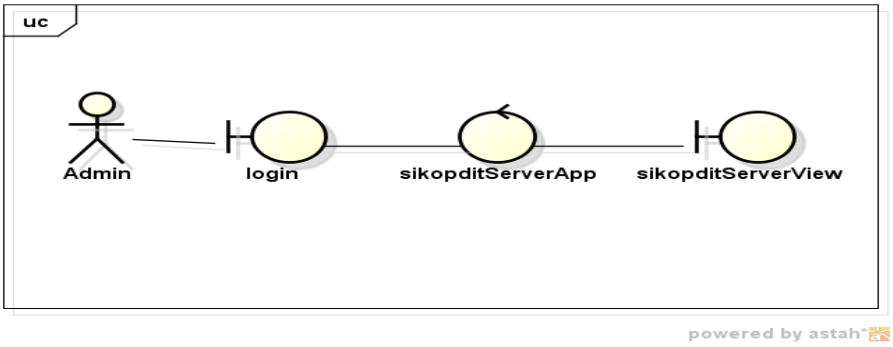


Gambar 3. 55 *Sequence Diagram* Lihat Histori Sinkronisasi

3.1.7.10 *Use Case* Tampil Data

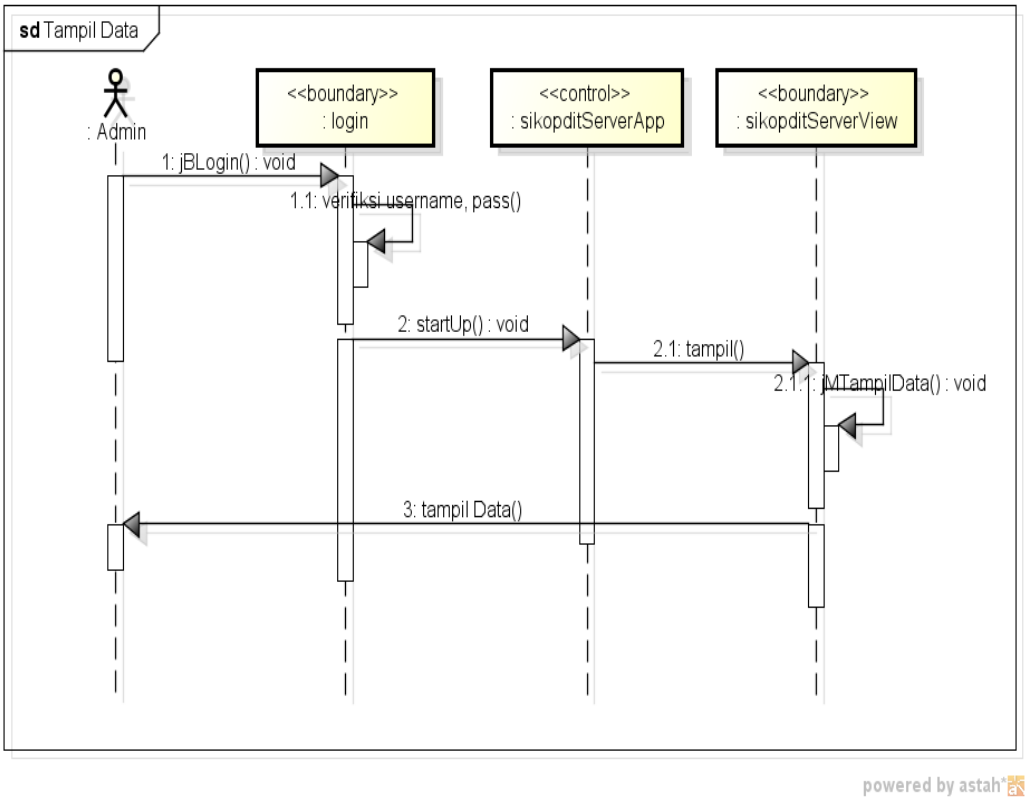
Hubungan interaksi antar objek *use case* login digambarkan sebagai berikut :

A. Kelas Analisis Tampil Data



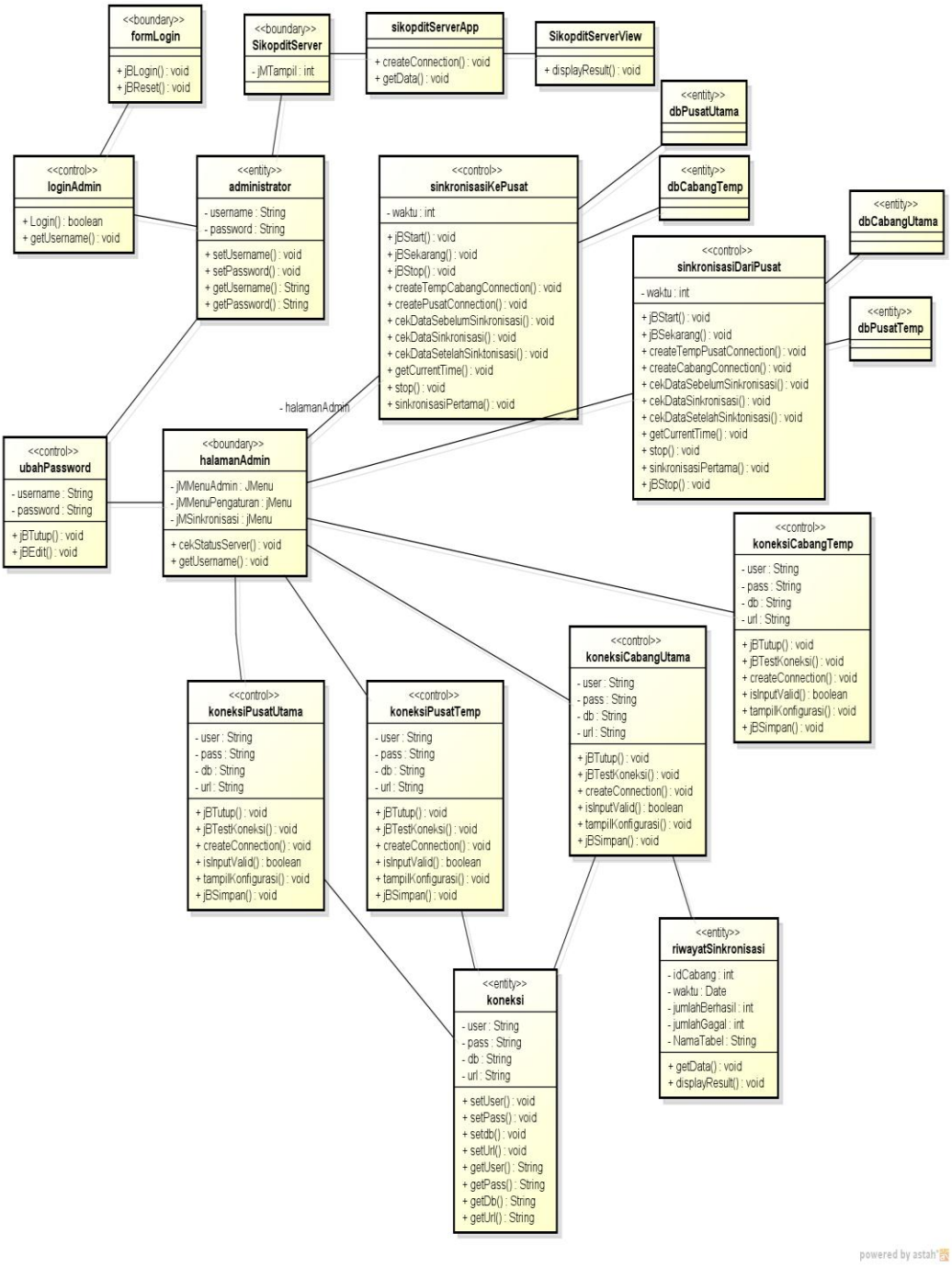
Gambar 3. 56 Kelas Analisis Tampil Data

B. *Sequence diagram* Tampil Data



Gambar 3. 57*Sequence diagram* Tampil Data

3.1.8 *Class Diagram*

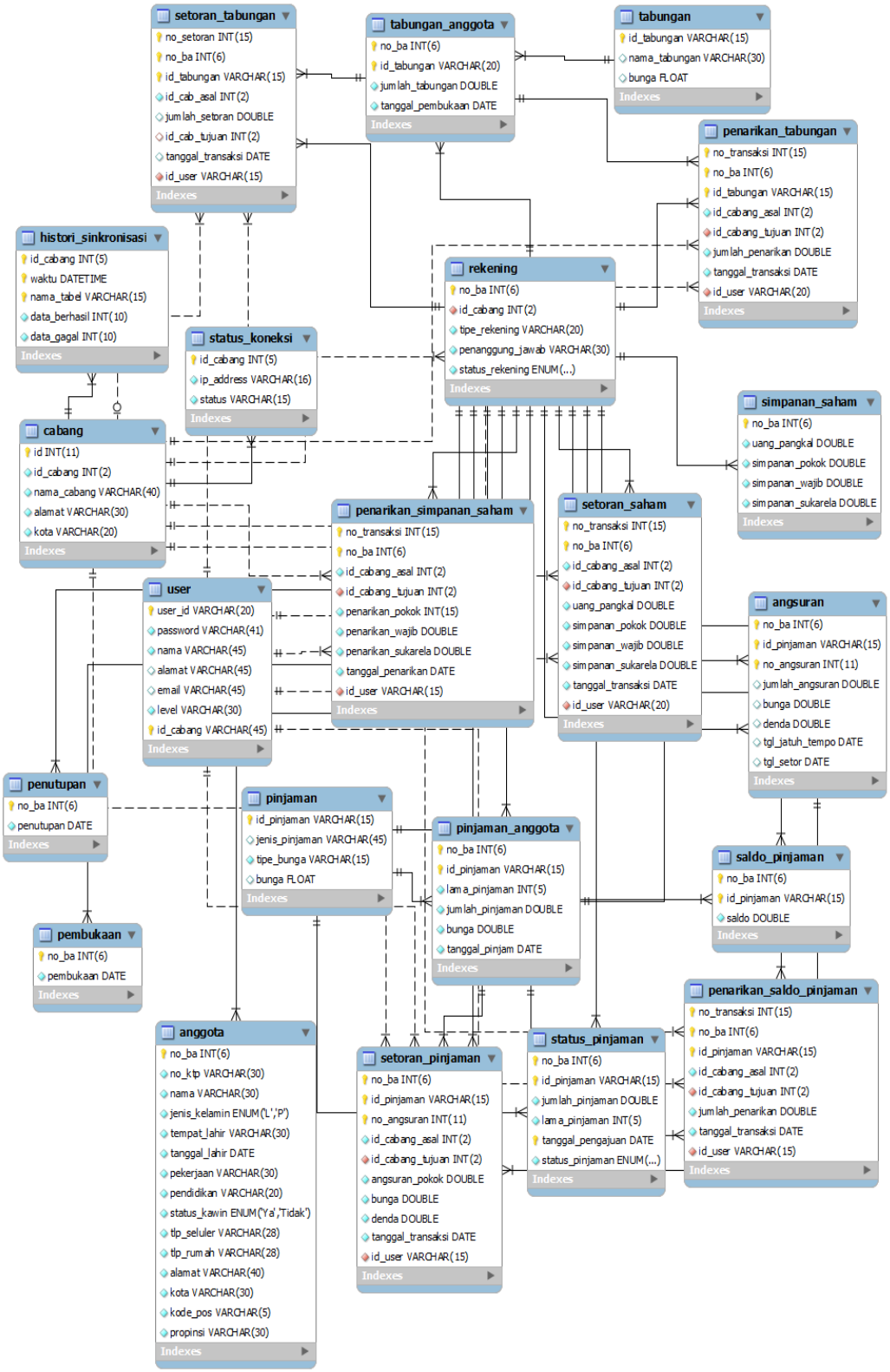


Gambar 3. 58 *Class Diagram Cabang*

3. 2 Perancangan

3.2.1 Skema Relasi Cabang dan Pusat

Skema relasi baik yang ada di *Database* cabang maupun yang ada di *Database* pusat sama. Sehingga seluruh tabel maupun f*ield*nya sama.



Gambar 3. 59 Skema Relasi Cabang dan Pusat

3.2.1.1 Struktur Tabel

1. Tabel Cabang

Tabel 3. 18 Tabel Cabang

Nama Field	Type	Size/Value	Keterangan
id	int	11	PK Auto Increment
id_cabang	int	2	
nama_cabang	varchar	30	
alamat_cabang	varchar	40	
kota	enum	20	

2. Tabel Rekening

Tabel 3. 19 Tabel Rekening

Nama Field	Type	Size/Value	Keterangan
no_ba	int	6	PK
id_cabang	int	2	FK ref. Cabang.id_cabang
tipe_rekening	varchar	20	
penanggung_jawab	varchar	30	
status_rekening	enum	Aktif, Tutup	

3. Tabel Anggota

Tabel 3. 20 Tabel Anggota

Nama Field	Type	Size/Value	Keterangan
no_ba	int	6	PK, FK ref rekening.no_ba
no_ktp	varchar	30	
nama	varchar	30	
jenis_kelamin	Enum	L,P	
tempat_lahir	varchar	30	
tanggal_lahir	date		
pekerjaan	varchar	30	
pendidikan	varchar	20	
status_kawin	enum	Ya, Tidak	
tlp_seluler	varchar	28	
tlp_rumah	varchar	28	

alamat	varchar	40	
kota	varchar	30	
kode_pos	int	5	
propinsi	varchar	30	

4. Tabel Pembukaan

Tabel 3. 21 Tabel Pembukaan

Nama Field	Type	Size/Value	Keterangan
no_ba	int	6	PK, FK ref rekening.no_ba
pembukaan	date		

5. Tabel Penutupan

Tabel 3. 22 Tabel Penutupan

Nama Field	Type	Size/Value	Keterangan
no_ba	int	6	PK, FK ref rekening.no_ba
penutupan	date		

6. Tabel Simpanan Saham

Tabel 3. 23 Tabel Simpanan Saham

Nama Field	Type	Size/Value	Keterangan
no_ba	int	6	PK, FK ref rekening.no_ba
uang_pangkal	double		
simpanan_pokok	double		
simpaan_wajib	double		
simpanan_sukarela	double		

7. Tabel Setoran Saham

Tabel 3. 24 Tabel Setoran Saham

Nama Field	Type	Size/Value	Keterangan

no_transaksi	int	15	PK,auto_inc
no_ba	int	6	FK ref rekening.no_ba
id_cabang_asal	int	2	
id_cabang_tujuan	int	2	FK ref cabang.id_cabang
uang_pangkal	double		
simpanan_pokok	double		
simpanan_wajib	double		
simpanan_sukarela	double		
tanggal_transaksi	date		
id_user	varchar	20	FK ref user.id_user

8. Tabel Penarikan Simpanan Saham

Tabel 3. 25 Tabel Penarikan Simpanan Saham

Nama Field	Type	Size/Value	Keterangan
no_transaksi	int	15	PK,auto_inc
no_ba	int	6	FK ref rekening.no_ba
id_cabang_asal	int	2	
id_cabang_tujuan	int	2	FK ref cabang.id_cabang
penarikan_pokok	double		
penarikan_wajib	double		
penarikan_sukarela	double		
tanggal_transaksi	date		
id_user	varchar	20	FK ref user.id_user

9. Tabel Tabungan

Tabel 3. 26 Tabel Tabungan

Nama Field	Type	Size/Value	Keterangan
id_tabungan	varchar	15	PK
nama_tabungan	varchar	30	
bunga	flaot		

10. Tabel Tabungan Anggota

Tabel 3. 27 Tabel Tabungan Anggota

Nama Field	Type	Size/Value	Keterangan
no_ba	int	6	PK, FK ref rekening.no_ba
id_tabungan	varchar	20	PK, FK tabungan.id_tabungan
tanggal_pembukaan	date		

11. Tabel Setoran Tabungan

Tabel 3. 28 Tabel Setoran Tabungan

Nama Field	Type	Size/Value	Keterangan
no_setoran	int	15	PK, auto_inc
no_ba	int	6	PK, FK ref rekening.no_ba
id_tabungan	varchar	15	FK,ref tabungan_anggota.id_tabungan
id_cab_asal	int	2	
id_cab_tujuan	int	2	FK ref cabang.id_cabang
jumlah_setoran	double		
tanggal_transaksi	date		
id_user	varchar	20	FK ref user.id_user

12. Tabel Penarikan Tabungan

Tabel 3. 29 Tabel Penarikan Tabungan

Nama Field	Type	Size/Value	Keterangan
no_tramsaksi	int	15	PK, auto_inc
no_ba	int	6	PK, FK ref rekening.no_ba
id_tabungan	varchar	15	FK,ref tabungan_anggota.id_tabungan
id_cab_asal	int	2	
id_cab_tujuan	int	2	FK ref cabang.id_cabang
jumlah_penarikan	double		
tanggal_transaksi	date		
id_user	varchar	20	FK ref user.id_user

13. Tabel Pinjaman

Tabel 3. 30 Tabel Pinjaman

Nama Field	Type	Size/Value	Keterangan
id_pinjaman	Varchar	15	PK
Jenis_pinjaman	Varchar	30	
Tipe_bunga	Varchar	15	
Bunga	Float		

14. Tabel Status Pinjaman

Tabel 3. 31 Tabel Status Pengajuan Pinjaman

Nama Field	Type	Size/Value	Keterangan
No_ba	int	15	PK, FK ref.rekening.no_ba
Id_pinjaman	int	6	FK ref pinjaman.id_pinjaman
Lama_pinjaman	int	6	
Tanggal_pinjaman	int	6	
jumlah_penarikan	double		
Status_pinjaman	date		

15. Tabel Pinjaman Anggota

Tabel 3. 32 Tabel Pinjaman Anggota

Nama Field	Type	Size/Value	Keterangan
no_ba	int	6	PK, FK ref rekening.no_ba
id_pinjaman	varchar	15	PK, FK ref pinjaman.id_pinjaman
lama_pinjaman	int	5	
jumlah_pinjaman	double		
bunga	double		
tanggal_pinjam	date		

16. Tabel Angsuran

Tabel 3. 33 Tabel Angsuran

Nama Field	Type	Size/Value	Keterangan
no_ba	int	6	PK, FK ref rekening.no_ba
id_pinjaman	varchar	15	PK, FK ref.pinjaman_anggota.id_pinjaman
lama_pinjaman	int	5	
jumlah_angsuran	double		
bunga	double		
denda	double		
tgl_jatuh_tempo	date		
tgl_setor	date		

17. Tabel Setoran Pinjaman

Tabel 3. 34 Tabel Setoran Pinjaman

Nama Field	Type	Size/Value	Keterangan
no_ba	int	6	PK, FK ref rekening.no_ba
id_pinjaman	varchar	15	PK, FK ref.pinjaman_anggota.id_pinjaman
id_cab_asal	int	2	
id_cab_tujuan	int	2	FK ref cabang.id_cabang
jumlah_penarikan	double		
angsuran_pokok	double		
bunga	double		
denda	doube		
tanggal_transaksi	date		
id_user	varchar	20	FK ref user.id_user

18. Tabel Saldo Pinjaman

Tabel 3. 35 Tabel Saldo Pinjaman

Nama Field	Type	Size/Value	Keterangan
no_ba	int	6	PK, FK ref rekening.no_ba
id_pinjaman	varchar	15	PK, FK ref.pinjaman_anggota.id_pinjaman
saldo	double		

19. Tabel Penarikan Saldo Pinjaman

Tabel 3. 36 Tabel Penarikan Saldo Pinjaman

Nama Field	Type	Size/Value	Keterangan
no_transaksi	int	15	PK,auto_inc
no_ba	int	6	FK ref rekening.no_ba
id_cabang_asal	int	6	FK ref cabang.id_cabang
id_cabang_tujuan	int	6	FK ref cabang.id_cabang
jumlah_penarikan	double		
tanggal_transaksi	date		
id_user	varchar	20	FK ref user.id_user

20. Tabel User

Tabel 3. 37 Tabel User

Nama Field	Type	Size/Value	Keterangan
id_user	varchar	20	PK
password	varchar	41	
nama	varchar	45	
alamat	varchar	45	
email	varchar	30	
level	varchar	15	
id_cabang	int	20	PK

21. Tabel Status Koneksi

Tabel 3. 38 Tabl Status Koneksi

Nama Field	Type	Size/Value	Keterangan
Id_cabang	int	2	PK
Ip_address	varchar	16	
status	varchar	15	

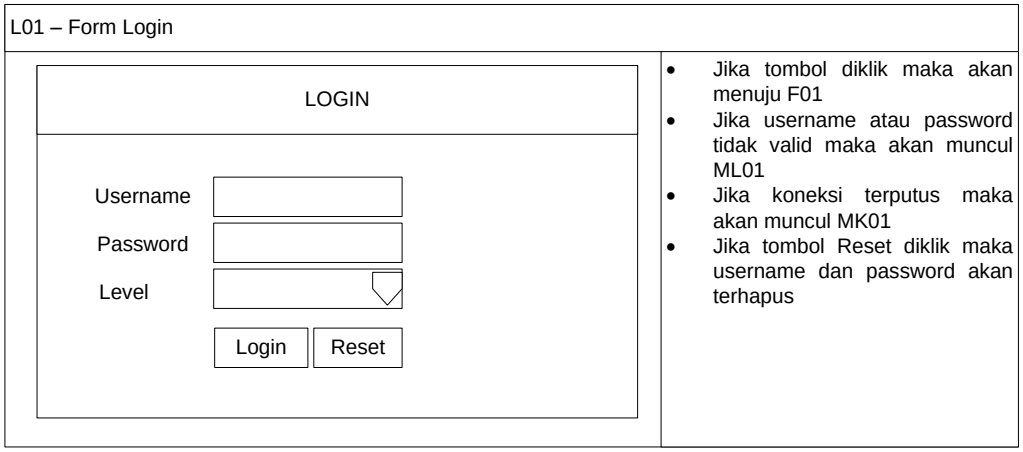
22. Tabel Histori Sinkronisasi

Tabel 3. 39 Tabel Histori Sinkronisasi

Nama Field	Type	Size/Value	Keterangan
Id_cabang	int	2	PK
waktu	datetime		PK
Nama_tabel	varchar	15	PK
Data_berhasil	int	10	
Dat_gagal	int	10	

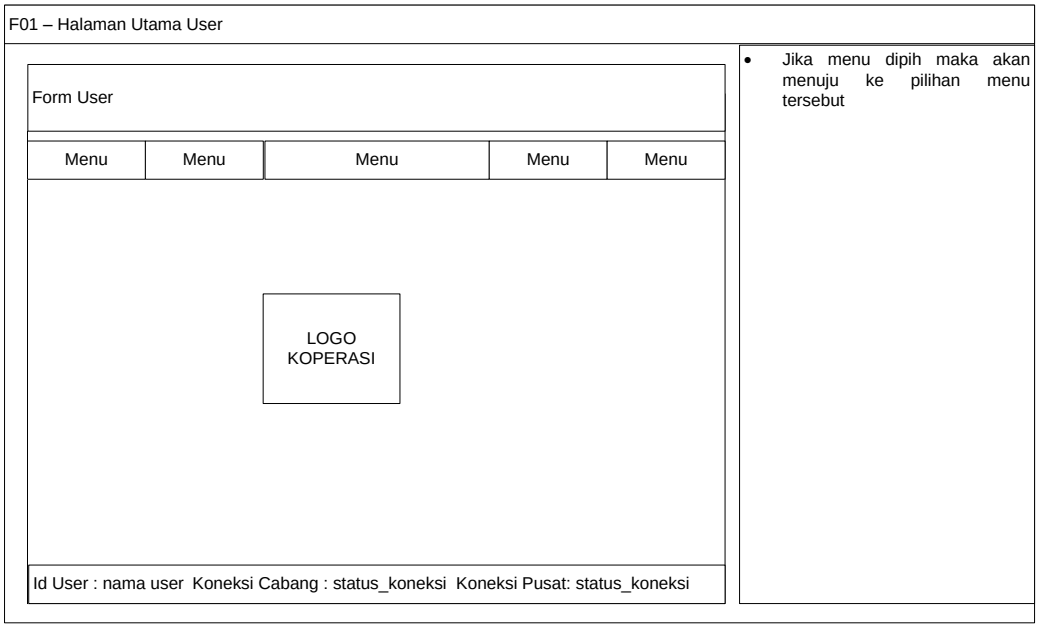
3.2.2 Perancangan Antar Muka Sistem

3.2.2.1 Login



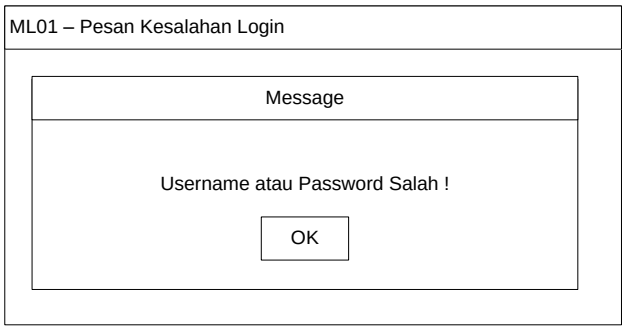
Gambar 3. 60Antar Muka Login

3.2.2.2 Halaman Utama Admin



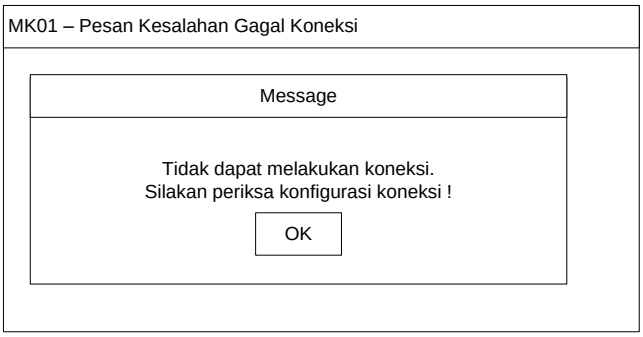
Gambar 3. 61 Antar Muka Halaman Utama User

3.2.2.3 Pesan Kesalahan Login



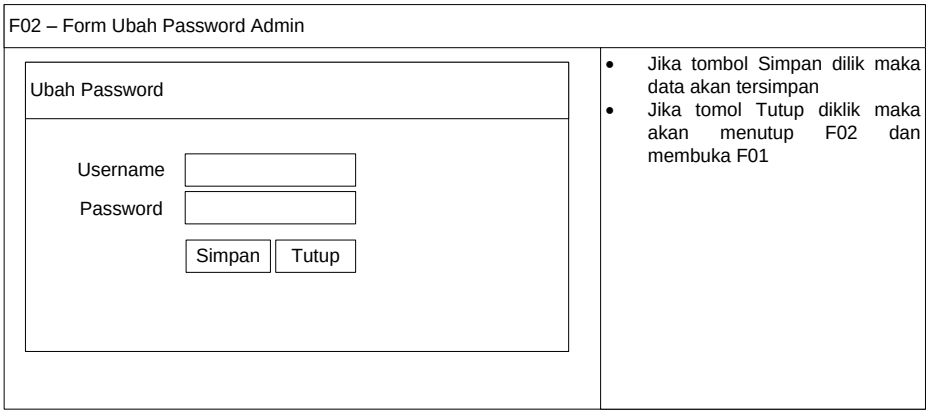
Gambar 3. 62 Antar Muka Pesan Kesalahan Login

3.2.2.4 Pesan Kesalahan Gagal Koneksi



Gambar 3. 63 Antar Muka Pesan Kesalahan Gagal Koneksi

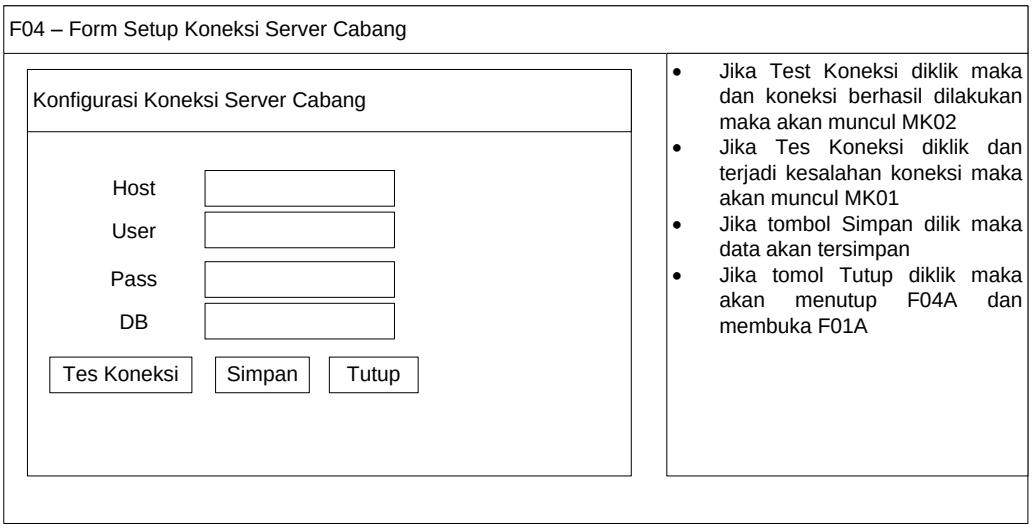
3.2.2.5 Form Ubah Password Admin



Gambar 3. 64 Antar Muka Form Ubah Password Admin

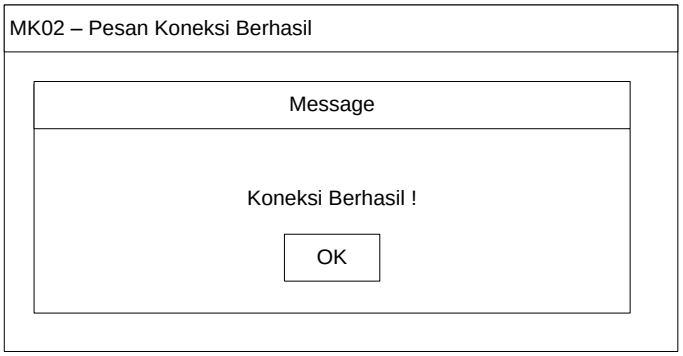
Gambar 3. 65 Antar Muka Form Pengolahan User

3.2.2.6 Form Setup Koneksi



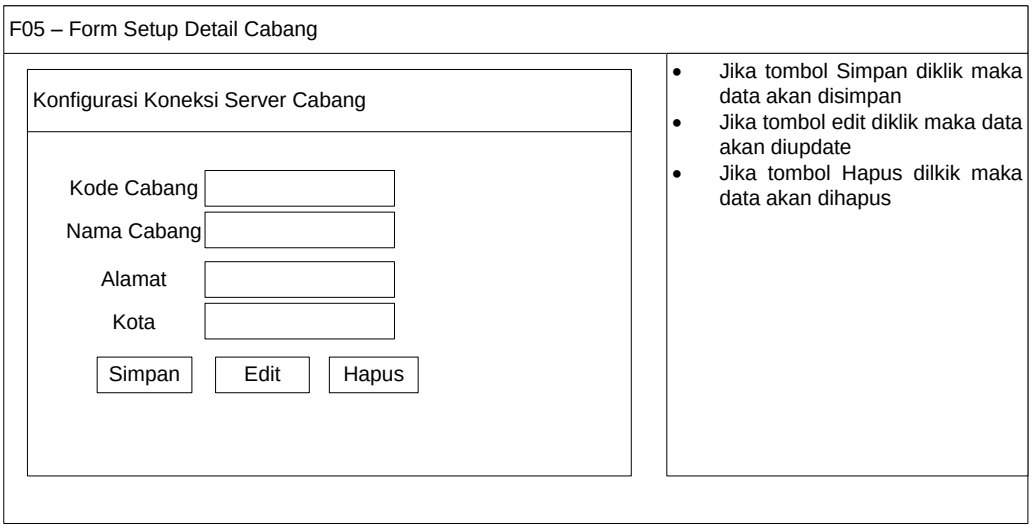
Gambar 3. 66 Antar Muka Form Setup Koneksi Cabang

3.2.2.7 Pesan Koneksi Berhasil



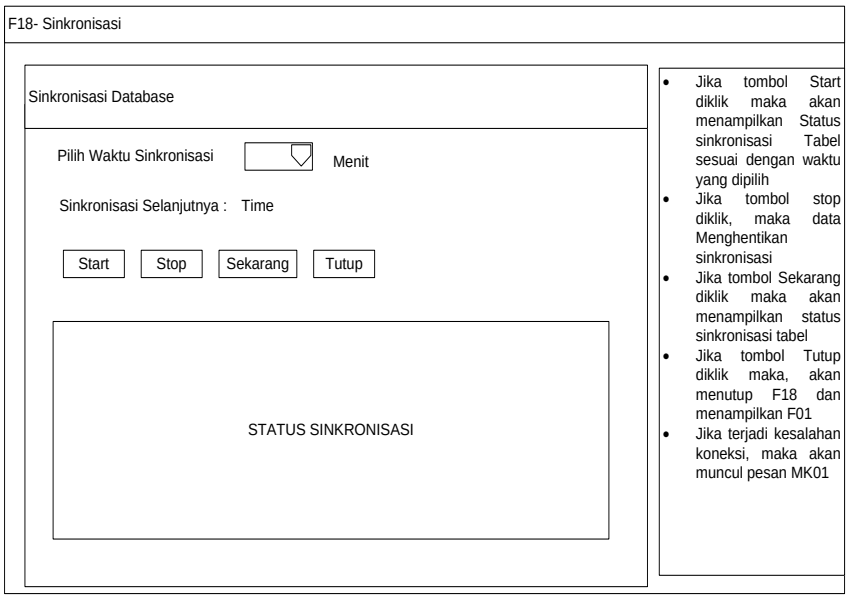
Gambar 3. 67 Antar Muka Pesan Koneksi Berhasil

3.2.2.8 Form Setup Detail Cabang



Gambar 3. 68 Antar Muka Form Setup Detail Cabang

3.2.2.9 Sinkronisasi



Gambar 3. 69 Antar Muka Sinkronisasi