BAB IV
IMPLEMENTASI DAN PENGUJIAN

Pada bab ini sistem yang telah dianalisis selanjutnya akan diimplemtasikan pada bahasa pemograman. Setelah diimplementasikan sistem akan diuji kemudian dilihat jika masih ada kekurangan untuk pengembangan sistem selanjutnya.

4.1 Implementasi

Setelah sistem dianalisis secara rinci maka tahap selnjutnya adalah melakukan implementasi sistem. Tujuan dari implemetasi sistem adalah untuk mengkonfimasi modul perancangan dengan para user sehingga dapat memberi masukan dalam pengembangan.

4.1.1 Kebutuhan Perangkat Keras

Pada pengujian kali ini kebutuhan perangkat keras dibagi menjadi dua, yaitu kebutuhan perangkat keras bagi server cabang dan kebutuhan perang keras bagi server pusat.

Khusus bagi server pusat, server akan menggunakan VPN *(Virtual Private Network).* Tujuannya ialah untuk menjamin bahwa server pusat selalu dalam keadaan *online.* Berikut adalah kebutuhannya:

Tabel 4. 1 Kebutuhan Perangkat Keras

Perangkat	Server Cabang	Server Pusat (VPN)	Server Pusat (Slave)
Processor	AMD Athlon X2	Intel Core 2 Duo	Intel Atom
RAM	2 GB DDR2	2 GB DDR2	1 GB DDR2
Harddisk	160 GB	160 GB	320 GB
Monitor	17”	16”	14”

4.1.2 Kebutuhan Perangkat Lunak

Perangkat lunak yang dibutuhkan untuk melakukan implementasi ialah sebagai berikut:

Tabel 4. 2 Implementasi Kebutuhan Perangkat Lunak

Perangkat Lunak	Server Cabang	Server Pusat (VPN)	Server Pusat (Slave)
Sistem Operasi	Windows 7	Windows Vista	Windows 7
Database Server	MySQL	MySQL	MySQL
JDK	JDK 5	-	-

4.1.3 Kebutuhan Jaringan Komputer

Kebutuhan jaringan komputer yang diperlukan untuk melakukan implementasi adalah:

Tabel 4. 3 Implementasi Kebutuhan Jaringan

Perangkat Jaringan	Client Cabang	Server Cabang	Server Pusat (VPN)
LAN	10/100 Mbps	10/100 Mbps	10/100 Mbps
Koneksi Internet	200 Kbps	312 Kbps	312 Kpbs
IP	Private IP	Dynamic IP	Dynaic IP
ISP	AHA	Speedy	Speedy

4.1.4 Implementasi Basis Data

Pembuatan *Database* dilakukan menggunakan DBMS MySQL. Adapun implementasi *Database* adalah sebagai beikut:

Tabel 4. 4 Implementasi Basis Data

No	Tabel	SQL
1	Cabang	CREATE TABLE IF NOT EXISTS `cabang` (`id` int(11) NOT NULL AUTO_INCREMENT, `id_cabang` int(2) NOT NULL, `nama_cabang` varchar(40) NOT NULL, `alamat` varchar(30) NOT NULL, `kota` varchar(20) NOT NULL, PRIMARY KEY (`id`), UNIQUE KEY `id_cabang` (`id_cabang`)) ENGINE=InnoDB DEFAULT

		CHARSET=latin1;
2	User	CREATE TABLE IF NOT EXISTS `user` (`user_id` varchar(20) NOT NULL, `password` varchar(41) NOT NULL, `nama` varchar(45) NOT NULL, `alamat` varchar(45) DEFAULT NULL, `email` varchar(45) DEFAULT NULL, `level` varchar(30) NOT NULL, `id_cabang` varchar(45) NOT NULL DEFAULT ‘’, PRIMARY KEY (`user_id`,`id_cabang`)) ENGINE=InnoDB DEFAULT CHARSET=latin1;
3	Rekening	CREATE TABLE IF NOT EXISTS `rekening` (`no_ba` int(6) NOT NULL, `id_cabang` int(2) NOT NULL, `tipe_rekening` varchar(20) NOT NULL, `penanggung_jawab` varchar(30) NOT NULL, `status_rekening` enum(‘Aktif’,’Tutup’’’) NOT NULL, PRIMARY KEY (`no_ba`)) ENGINE=InnoDB DEFAULT CHARSET=latin1;
4	Anggota	CREATE TABLE IF NOT EXISTS `anggota` (

		`no_ba` int(6) NOT NULL, `no_ktp` varchar(30) NOT NULL, `nama` varchar(30) NOT NULL, `jenis_kelamin` enum('L','P') NOT NULL, `tempat_lahir` varchar(30) NOT NULL, `tanggal_lahir` date NOT NULL, `pekerjaan` varchar(30) NOT NULL, `pendidikan` varchar(20) NOT NULL, `status_kawin` enum('Ya','Tidak') NOT NULL, `tlp_seluler` varchar(28) NOT NULL, `tlp_rumah` varchar(28) NOT NULL, `alamat` varchar(40) NOT NULL, `kota` varchar(30) NOT NULL, `kode_pos` varchar(5) NOT NULL, `propinsi` varchar(30) NOT NULL, PRIMARY KEY (`no_ba`)) ENGINE=InnoDB DEFAULT CHARSET=latin1;
5	Pembukaan	CREATE TABLE IF NOT EXISTS `pembukaan` (`no_ba` int(6) NOT NULL, `pembukaan` date NOT NULL, PRIMARY KEY (`no_ba`)) ENGINE=InnoDB DEFAULT CHARSET=latin1;

		ALTER TABLE `pembukaan` ADD CONSTRAINT `pembukaan_ibfk_1` FOREIGN KEY (`no_ba`) REFERENCES `rekening` (`no_ba`) ON UPDATE CASCADE;
6	Penutupan	CREATE TABLE IF NOT EXISTS `penutupan` (`no_ba` int(6) NOT NULL, `penutupan` date NOT NULL, PRIMARY KEY (`no_ba`)) ENGINE=InnoDB DEFAULT CHARSET=latin1; ALTER TABLE `penutupan` ADD CONSTRAINT `penutupan_ibfk_1` FOREIGN KEY (`no_ba`) REFERENCES `rekening` (`no_ba`) ON UPDATE CASCADE;
7	Simpanan_saham	CREATE TABLE IF NOT EXISTS `simpanan_saham` (`no_ba` int(6) NOT NULL, `uang_pangkal` double NOT NULL, `simpanan_pokok` double NOT NULL, `simpanan_wajib` double NOT NULL, `simpanan_sukarela` double NOT NULL, PRIMARY KEY (`no_ba`)) ENGINE=InnoDB DEFAULT CHARSET=latin1; ALTER TABLE `simpanan_saham`

		ADD CONSTRAINT `simpanan_saham_ibfk_1` FOREIGN KEY (`no_ba`) REFERENCES `rekening` (`no_ba`) ON UPDATE CASCADE;
8	Setoran_saham	CREATE TABLE IF NOT EXISTS `setoran_saham` (`no_transaksi` int(15) NOT NULL AUTO_INCREMENT, `no_ba` int(6) NOT NULL, `id_cabang_asal` int(2) NOT NULL, `id_cabang_tujuan` int(2) NOT NULL, `uang_pangkal` double NOT NULL, `simpanan_pokok` double NOT NULL, `simpanan_wajib` double NOT NULL, `simpanan_sukarela` double NOT NULL, `tanggal_transaksi` date NOT NULL, `id_user` varchar(20) NOT NULL, PRIMARY KEY (`no_transaksi`,`no_ba`,`id_cabang_asal`), KEY `no_ba` (`no_ba`), KEY `id_cabang_asal` (`id_cabang_asal`), KEY `id_user` (`id_user`)) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ; ALTER TABLE `setoran_saham`

		ADD CONSTRAINT `setoran_saham_ibfk_1` FOREIGN KEY (`no_ba`) REFERENCES `rekening` (`no_ba`) ON UPDATE CASCADE, ADD CONSTRAINT `setoran_saham_ibfk_3` FOREIGN KEY (`id_user`) REFERENCES `user` (`user_id`) ON UPDATE CASCADE, ADD CONSTRAINT `setoran_saham_ibfk_4` FOREIGN KEY (`id_cabang_asal`) REFERENCES `cabang` (`id_cabang`) ON UPDATE CASCADE;
9	Penarikan_simpanan_saham	CREATE TABLE IF NOT EXISTS `penarikan_simpanan_saham` (`no_transaksi` int(15) NOT NULL AUTO_INCREMENT, `no_ba` int(6) NOT NULL, `id_cabang_asal` int(2) NOT NULL, `id_cabang_tujuan` int(2) NOT NULL, `penarikan_pokok` int(15) NOT NULL, `penarikan_wajib` double NOT NULL, `penarikan_sukarela` double NOT NULL, `tanggal_penarikan` date NOT NULL, `id_user` varchar(15) NOT NULL, PRIMARY KEY (`no_transaksi`,`no_ba`), KEY `no_ba` (`no_ba`), KEY `id_cabang_asal` (`id_cabang_asal`,`id_cabang_t

		ujuan`), KEY `id_cabang_tujuan` (`id_cabang_tujuan`), KEY `id_user` (`id_user`) ) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ;  ALTER TABLE `penarikan_simpanan_saham` ADD CONSTRAINT `penarikan_simpanan_saham_ibfk_2` FOREIGN KEY (`id_user`) REFERENCES `user` (`user_id`) ON UPDATE CASCADE, ADD CONSTRAINT `penarikan_simpanan_saham_ibfk_3` FOREIGN KEY (`no_ba`) REFERENCES `rekening` (`no_ba`) ON UPDATE CASCADE, ADD CONSTRAINT `penarikan_simpanan_saham_ibfk_4` FOREIGN KEY (`id_cabang_asal`) REFERENCES `cabang` (`id_cabang`) ON UPDATE CASCADE;
10	Tabungan	CREATE TABLE IF NOT EXISTS `penarikan_tabungan` (`no_transaksi` int(15) NOT NULL AUTO_INCREMENT, `no_ba` int(6) NOT NULL, `id_tabungan` varchar(15) NOT NULL, `id_cabang_asal` int(2) NOT NULL, `id_cabang_tujuan` int(2) NOT NULL, `jumlah_penarikan` double NOT NULL,

		`tanggal_transaksi` date NOT NULL, `id_user` varchar(20) NOT NULL, PRIMARY KEY (`no_transaksi`,`no_ba`,`id_tabungan`), KEY `no_ba` (`no_ba`), KEY `id_cabang_asal` (`id_cabang_asal`), KEY `id_tabungan` (`id_tabungan`), KEY `id_user` (`id_user`)) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ;
11	Pinjaman	CREATE TABLE IF NOT EXISTS `pinjaman` (`id_pinjaman` varchar(15) NOT NULL, `jenis_pinjaman` varchar(45) DEFAULT NULL, `tipe_bunga` varchar(15) NOT NULL, `bunga` float DEFAULT NULL, PRIMARY KEY (`id_pinjaman`)) ENGINE=InnoDB DEFAULT CHARSET=latin1;
12	Tabungan_anggota	CREATE TABLE IF NOT EXISTS `tabungan_anggota` (`no_ba` int(6) NOT NULL, `id_tabungan` varchar(20) NOT NULL, `jumlah_tabungan` double NOT NULL, `tanggal_pembukaan` date NOT NULL,

		PRIMARY KEY (`no_ba`,`id_tabungan`), KEY `id_tabungan` (`id_tabungan`)) ENGINE=InnoDB DEFAULT CHARSET=latin1; ALTER TABLE `tabungan_anggota` ADD CONSTRAINT `tabungan_anggota_ibfk_1` FOREIGN KEY (`no_ba`) REFERENCES `rekening` (`no_ba`) ON UPDATE CASCADE, ADD CONSTRAINT `tabungan_anggota_ibfk_2` FOREIGN KEY (`id_tabungan`) REFERENCES `tabungan` (`id_tabungan`) ON UPDATE CASCADE;
13	Setoran_tabungan	CREATE TABLE IF NOT EXISTS `setoran_tabungan` (`no_setoran` int(15) NOT NULL AUTO_INCREMENT, `no_ba` int(6) NOT NULL, `id_tabungan` varchar(15) NOT NULL, `id_cab_asal` int(2) NOT NULL, `jumlah_setoran` double DEFAULT NULL, `id_cab_tujuan` int(2) DEFAULT NULL, `tanggal_transaksi` date DEFAULT NULL, `id_user` varchar(15) NOT NULL, PRIMARY KEY (`no_setoran`,`no_ba`,`id_tabungan`),

		KEY `id_cab_asal` (`id_cab_asal`), KEY `no_ba` (`no_ba`), KEY `id_tabungan` (`id_tabungan`), KEY `id_user` (`id_user`)) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ; ALTER TABLE `setoran_tabungan` ADD CONSTRAINT `setoran_tabungan_ibfk_1` FOREIGN KEY (`no_ba`) REFERENCES `rekening` (`no_ba`) ON UPDATE CASCADE, ADD CONSTRAINT `setoran_tabungan_ibfk_2` FOREIGN KEY (`id_tabungan`) REFERENCES `tabungan` (`id_tabungan`) ON UPDATE CASCADE, ADD CONSTRAINT `setoran_tabungan_ibfk_4` FOREIGN KEY (`id_user`) REFERENCES `user` (`user_id`) ON UPDATE CASCADE, ADD CONSTRAINT `setoran_tabungan_ibfk_5` FOREIGN KEY (`id_cab_asal`) REFERENCES `cabang` (`id_cabang`) ON UPDATE CASCADE;
14	Penarikan_tabungan	CREATE TABLE IF NOT EXISTS `penarikan_tabungan` (`no_transaksi` int(15) NOT NULL AUTO_INCREMENT, `no_ba` int(6) NOT NULL, `id_tabungan` varchar(15) NOT NULL,

		`id_cabang_asal` int(2) NOT NULL, `id_cabang_tujuan` int(2) NOT NULL, `jumlah_penarikan` double NOT NULL, `tanggal_transaksi` date NOT NULL, `id_user` varchar(20) NOT NULL, PRIMARY KEY (`no_transaksi`,`no_ba`,`id_tabungan`), KEY `no_ba` (`no_ba`), KEY `id_cabang_asal` (`id_cabang_asal`), KEY `id_tabungan` (`id_tabungan`), KEY `id_user` (`id_user`)) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ; ALTER TABLE `penarikan_tabungan` ADD CONSTRAINT `penarikan_tabungan_ibfk_1` FOREIGN KEY (`no_ba`) REFERENCES `rekening` (`no_ba`) ON UPDATE CASCADE, ADD CONSTRAINT `penarikan_tabungan_ibfk_2` FOREIGN KEY (`id_tabungan`) REFERENCES `tabungan` (`id_tabungan`) ON UPDATE CASCADE, ADD CONSTRAINT `penarikan_tabungan_ibfk_3` FOREIGN KEY (`id_user`) REFERENCES `user` (`user_id`)

		ON UPDATE CASCADE, ADD CONSTRAINT `penarikan_tabungan_ibfk_4` FOREIGN KEY (`id_cabang_asal`) REFERENCES `cabang` (`id_cabang`) ON UPDATE CASCADE;
15	Status_pinjaman	CREATE TABLE IF NOT EXISTS `status_pinjaman` (`no_ba` int(6) NOT NULL, `id_pinjaman` varchar(15) NOT NULL, `jumlah_pinjaman` double NOT NULL, `lama_pinjaman` int(5) NOT NULL, `tanggal_pengajuan` date NOT NULL, `status_pinjaman` enum('Menunggu','Diterima','Ditolak') NOT NULL, PRIMARY KEY (`no_ba`,`id_pinjaman`,`tanggal_pengajuan`), KEY `id_pinjaman` (`id_pinjaman`)) ENGINE=InnoDB DEFAULT CHARSET=latin1; ALTER TABLE `status_pinjaman` ADD CONSTRAINT `status_pinjaman_ibfk_1` FOREIGN KEY (`no_ba`) REFERENCES `rekening` (`no_ba`) ON UPDATE CASCADE, ADD CONSTRAINT `status_pinjaman_ibfk_2` FOREIGN KEY (`id_pinjaman`) REFERENCES `pinjaman` (`id_pinjaman`) ON UPDATE

		CASCADE;
16	Pinjaman_anggota	CREATE TABLE IF NOT EXISTS `pinjaman_anggota` (`no_ba` int(6) NOT NULL, `id_pinjaman` varchar(15) NOT NULL, `lama_pinjaman` int(5) NOT NULL, `jumlah_pinjaman` double NOT NULL, `bunga` double NOT NULL, `tanggal_pinjam` date NOT NULL, PRIMARY KEY (`no_ba`,`id_pinjaman`), KEY `id_pinjaman` (`id_pinjaman`)) ENGINE=InnoDB DEFAULT CHARSET=latin1; ALTER TABLE `pinjaman_anggota` ADD CONSTRAINT `pinjaman_anggota_ibfk_1` FOREIGN KEY (`no_ba`) REFERENCES `rekening` (`no_ba`) ON UPDATE CASCADE, ADD CONSTRAINT `pinjaman_anggota_ibfk_2` FOREIGN KEY (`id_pinjaman`) REFERENCES `pinjaman` (`id_pinjaman`) ON UPDATE CASCADE;
17	Saldo_pinjaman	CREATE TABLE IF NOT EXISTS `saldo_pinjaman` (`no_ba` int(6) NOT NULL, `id_pinjaman` varchar(15) NOT NULL,

		`saldo` double NOT NULL, PRIMARY KEY (`no_ba`,`id_pinjaman`), KEY `id_pinjaman` (`id_pinjaman`)) ENGINE=InnoDB DEFAULT CHARSET=latin1; ALTER TABLE `saldo_pinjaman` ADD CONSTRAINT `saldo_pinjaman_ibfk_1` FOREIGN KEY (`no_ba`) REFERENCES `rekening` (`no_ba`) ON UPDATE CASCADE, ADD CONSTRAINT `saldo_pinjaman_ibfk_2` FOREIGN KEY (`id_pinjaman`) REFERENCES `pinjaman` (`id_pinjaman`) ON UPDATE CASCADE;
18	Penarikan_saldo_pinjaman	CREATE TABLE IF NOT EXISTS `penarikan_saldo_pinjaman` (`no_transaksi` int(15) NOT NULL AUTO_INCREMENT, `no_ba` int(6) NOT NULL, `id_pinjaman` varchar(15) NOT NULL, `id_cabang_asal` int(2) NOT NULL, `id_cabang_tujuan` int(2) NOT NULL, `jumlah_penarikan` double NOT NULL, `tanggal_transaksi` date NOT NULL, `id_user` varchar(15) NOT NULL, PRIMARY KEY (`no_transaksi`,`no_ba`,`id_pi

		njaman`), KEY `no_ba` (`no_ba`), KEY `id_user` (`id_user`), KEY `id_pinjaman` (`id_pinjaman`), KEY `id_cabang_asal` (`id_cabang_asal`) ) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ;  ALTER TABLE `penarikan_saldo_pinjaman` ADD CONSTRAINT `penarikan_saldo_pinjaman_ibfk_1` FOREIGN KEY (`no_ba`) REFERENCES `rekening` (`no_ba`) ON UPDATE CASCADE, ADD CONSTRAINT `penarikan_saldo_pinjaman_ibfk_3` FOREIGN KEY (`id_user`) REFERENCES `user` (`user_id`) ON UPDATE CASCADE, ADD CONSTRAINT `penarikan_saldo_pinjaman_ibfk_4` FOREIGN KEY (`id_pinjaman`) REFERENCES `pinjaman` (`id_pinjaman`) ON UPDATE CASCADE, ADD CONSTRAINT `penarikan_saldo_pinjaman_ibfk_5` FOREIGN KEY (`id_cabang_asal`) REFERENCES `cabang` (`id_cabang`) ON UPDATE CASCADE;
19	Angsuran	CREATE TABLE IF NOT EXISTS `angsuran` (`no_ba` int(6) NOT NULL, `id_pinjaman` varchar(15) NOT NULL DEFAULT ‘’,

		`no_angsuran` int(11) NOT NULL, `jumlah_angsuran` double DEFAULT NULL, `bunga` double DEFAULT NULL, `denda` double DEFAULT NULL, `tgl_jatuh_tempo` date DEFAULT NULL, `tgl_setor` date DEFAULT NULL, PRIMARY KEY (`no_ba`,`id_pinjaman`,`no_angsuran`), KEY `id_pinjaman` (`id_pinjaman`), KEY `no_angsuran` (`no_angsuran`)) ENGINE=InnoDB DEFAULT CHARSET=latin1; ALTER TABLE `angsuran` ADD CONSTRAINT `angsuran_ibfk_1` FOREIGN KEY (`no_ba`) REFERENCES `rekening` (`no_ba`) ON UPDATE CASCADE, ADD CONSTRAINT `angsuran_ibfk_2` FOREIGN KEY (`id_pinjaman`) REFERENCES `pinjaman` (`id_pinjaman`) ON UPDATE CASCADE;
20	Setoran_pinjaman	CREATE TABLE IF NOT EXISTS `setoran_pinjaman` (`no_ba` int(6) NOT NULL, `id_pinjaman` varchar(15) NOT NULL, `no_angsuran` int(11) NOT NULL,

		`id_cabang_asal` int(2) NOT NULL, `id_cabang_tujuan` int(2) NOT NULL, `angsuran_pokok` double NOT NULL, `bunga` double NOT NULL, `denda` double NOT NULL, `tanggal_transaksi` date NOT NULL, `id_user` varchar(15) NOT NULL, PRIMARY KEY (`no_ba`,`id_pinjaman`,`no_angsuran`), KEY `id_cabang_asal` (`id_cabang_asal`), KEY `id_pinjaman` (`id_pinjaman`), KEY `no_angsuran` (`no_angsuran`), KEY `id_user` (`id_user`)) ENGINE=InnoDB DEFAULT CHARSET=latin1; ALTER TABLE `setoran_pinjaman` ADD CONSTRAINT `setoran_pinjaman_ibfk_1` FOREIGN KEY (`no_ba`) REFERENCES `rekening` (`no_ba`) ON UPDATE CASCADE, ADD CONSTRAINT `setoran_pinjaman_ibfk_2` FOREIGN KEY (`id_pinjaman`) REFERENCES `pinjaman` (`id_pinjaman`) ON UPDATE CASCADE, ADD CONSTRAINT `setoran_pinjaman_ibfk_3`

		FOREIGN KEY (`no_angsuran`) REFERENCES `angsuran` (`no_angsuran`) ON UPDATE CASCADE, ADD CONSTRAINT `setoran_pinjaman_ibfk_5` FOREIGN KEY (`id_user`) REFERENCES `user` (`user_id`) ON UPDATE CASCADE, ADD CONSTRAINT `setoran_pinjaman_ibfk_6` FOREIGN KEY (`id_cabang_asal`) REFERENCES `cabang` (`id_cabang`) ON UPDATE CASCADE;
21	Status Koneksi	CREATE TABLE IF NOT EXISTS `status_koneksi` (`id_cabang` int(5) NOT NULL, `ip_address` varchar(16) NOT NULL, `status` varchar(15) NOT NULL, PRIMARY KEY (`id_cabang`)) ENGINE=InnoDB DEFAULT CHARSET=latin1; ALTER TABLE `status_koneksi` ADD CONSTRAINT `status_koneksi_ibfk_1` FOREIGN KEY (`id_cabang`) REFERENCES `cabang` (`id_cabang`) ON DELETE CASCADE ON UPDATE CASCADE;
22	Histori Sinkronisasi	CREATE TABLE IF NOT EXISTS `histori_sinkronisasi` (`id_cabang` int(5) NOT NULL, `waktu` datetime NOT NULL, `nama_tabel` varchar(15) NOT NULL, `data_berhasil` int(10) NOT

		NULL, `data_gagal` int(10) NOT NULL, PRIMARY KEY (`id_cabang`,`waktu`,`nama_tabel`)) ENGINE=InnoDB DEFAULT CHARSET=latin1; ALTER TABLE `histori_sinkronisasi` ADD CONSTRAINT `histori_sinkronisasi_ibfk_1` FOREIGN KEY (`id_cabang`) REFERENCES `cabang` (`id_cabang`) ON DELETE CASCADE ON UPDATE CASCADE;

4.1.5 Implementasi VPN

Implementasi VPN dilakukan untuk mengaplikasikan konsep VPN pada transaksi antar jaringan komputer di peneletian ini. Pada penelitian kali ini akan digunakan tiga buah komputer. Satu sebagau server pusat dan dua sebagai server cabang. Berikut adalah adalah penjelasannya:

	VPN Server	VPN Client
ISP	Speedy	AHA
Jenis IP	Dynamic	Private
IP Address	114.79.33.193	

4.1.6 Implementasi Sinkronisasi *Database* Cabang Ke Pusat

Sinkronisasi *Database* cabang ke pusat dilakukan menggunakan dua buah *Database*. Yaitu *Database* temporer yang ada dikantor cabang dan *Database* cabang yang ada di kanto pusat.

Implementasi sinkronisasi *Database* cabang ke pusat dilakukan menggunkan bahasa pemograman java, berikut adalah algoritmanya :

1. Buat koneksi ke *Database* pusat dan cabang menggunakan konektor JDBC.
2. Lakukan pengecekan data ke *Database* temporer yang ada di kantor cabang. Kemudian hitung jumlah data yang akan disinkronisasikan.
3. Lakukan pengecekan data ke *Database* utana yang ada dikantor pusat, kemudian hitung jumlah data dari setiap tabel.
4. Cek apakah data sudah ada yang sama di *Database* kantor pusat.
5. Jika data sudah ada, maka lakukan *update*.
6. Jika data tidak ada, lakukan *inesert* ke *Database* pusat.
7. Jika berhasil, *hapus* data yang ada di *Database* temporer cabang dan lakukan *commit.*
8. Jika gagal, lakukan *rollback* pada *Database* temporer di kantor cabang.

4.1.7 Implementasi Sinkronisasi *Database* Pusat Ke Cabang

Sinkronisasi *Database* pusat ke cabang dilakukan menggunakan dua buah *Database.* Yaitu *Database* temporer yang ada dikantor pusat dan *Database* utama yang ada di kanto cabang.

Implementasi sinkronisasi *Database* pusat ke cabang dilakukan menggunkan bahasa pemograman java, berikut adalah algoritmanya :

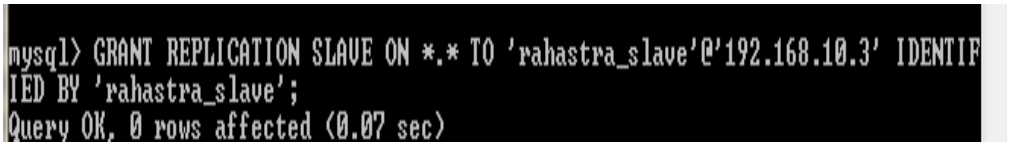
1. Buat koneksi ke *Database* pusat dan cabang menggunakan konektor JDBC.
2. Lakukan pengecekan data ke *Database* temporer yang ada di kantor pusat, berdasekan id cabang. Kemudian hitung jumlah data yang akan disinkronisasikan.
3. Lakukan pengecekan data di *Database* temporer yang ada dikantor pusat berdasarkan id cabang, kemudian hitung jumlah data dari setiap tabel.
4. Cek apakah data sudah ada yang sama di *Database* kantor cabang.
5. Jika data sudah ada, maka lakukan *update.*
6. Jika data tidak ada, lakukan *inesert* ke *Database* pusat.
7. Jika berhasil, *hapus* data yang ada di *Database* temporer pusat dan lakukan *commit.*
8. Jika gagal, lakukan *rollback* pada *Database* temporer di kantor pusat.

4.1.8 Implementasi Replikasi *Database* Pusat

Implementasi replikasi *Database* pusat menggunakan fasilitas replikasi bawaan MySQL. Untuk itu perlu dilakukan konfigurasi di *server* master dan *server slave.* Berikut adalah konfigurasinya :

4.1.8.1 Konfigurasi Pada *Database* Server Master

1 Buat account *slave* pada komputer *master:*



Gambar 4. 1 Membuat *account slave* pada komputer master

2 Ubah [wampmysqld] yang ada di file my.ini pada sisi server :

[wampmysqld]

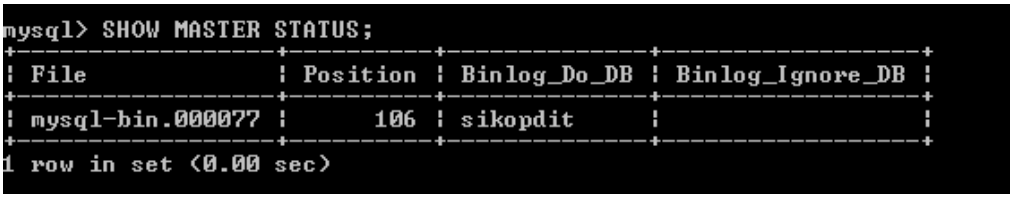
log

log-bin = mysql-log-bin

binlog-do-db = sikopdit

server-id = 1

3 Cek status master:



Gambar 4. 2 Show master status

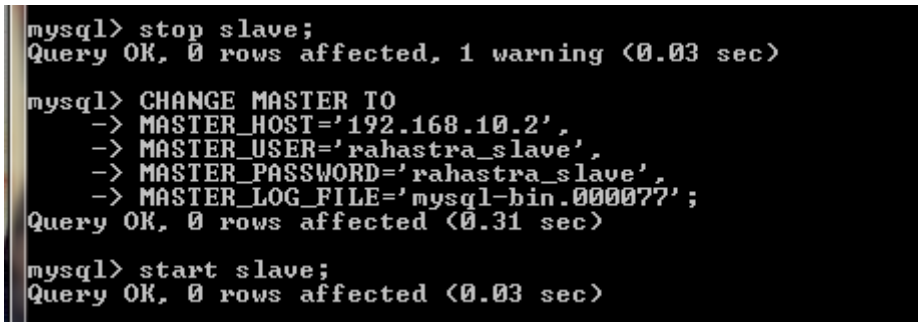
4.1.8.2 Konfigurasi Pada *Database* Server Slave

1 Ubah [wampmysqld] yang ada di file my.ini pada sisi server :

Server-id = 2

Replicate-do-db=sikopdit

2 Jalankan query berikut:



Gambar 4. 3 Konfigurasi di slave

4.1.9 Implementasi Antar Muka

Implementasi antar muka diambil dari setiap halaman program yang diakses oleh user. Berikut adalah uraiannya:

4.1.9.1 Implementasi Antar Muka Login

Antar muka login ialah antar muka yang terdapat pada saat aplikasi pertama kali dijalankan. Antar muka login dibuat untuk memudahkan user mengakses halaman yang dituju. Berikut penjelasannya:

Tabel 4. 5 Antar Muka Login

Menu	Deskripsi
Login	Untuk login user
Reset	Mengembalikan nilai ke keadaan semula (kosong)

4.1.9.2 Implementasi Antar Muka Halaman Admin

Antar muka halaman admin ditunjukan bagi admin yang mengelola sistem. Berikut adalah penjelasannya:

Tabel 4. 6 Antar Muka Halaman Admin

Menu	Deskripsi
Admin	Menu ini bersisi sub menu ubah password, logout dan keluar
Ubah password	Berfungsi untuk merubah password admin
Log out	Berfungsi untuk keluar dari halaman admin dan akan menuju ke halaman login
Keluar	Berfungsi untuk keluar dari sistem
Sinkronisasi	Menu ini berisi sub menu siknronisasi ke pusat dan sinkronisasi dari pusat
Sinbkronisasi ke pusat	Berfungsi untuk melakukan sinkronisasi *Database* dari *Database* cabang ke *Database*

	pusat
Sinkronisasi dari pusat	Berfungsi untuk melaukan sinkronisasi dari *Database* pusat ke *Database* cabang
Pengaturan	Menu ini berisi sub menu pengaturan cabang, koneksi *Database*pusat dan koneksi *Database* cabang.
Pengaturan cabang	Berfungsi untuk mengatur cabang
Koneksi *Database* pusat	Berisi sub menu pengaturan *Database* utama dan pengaturan *Database* temp
Pengaturan *Database* utama (pusat)	Berfungsi untuk mengatur koneksi *Database* utama pusat
Pengaturan *Database* temp (pusat)	Berfungsi untuk mengatur koneksi *Database* temporer pusat
Koneksi *Database* cabang	Berisi sub menu pengaturan *Database* utama dan pengaturan *Database* temp
Pengaturan *Database* utama (cabang)	Berfungsi untuk mengatur koneksi *Database* utama cabang
Pengaturan *Database* temp (cabang)	Berfungsi untuk mengatur koneksi *Database* temporer cabang
Bantuan	Berisi sub menu tentang kami
Tentang kami	Berisi informasi mengenai pembuat aplikasi
Pengguna	Berfungsi untuk menampilkan infotmsi user yang aktif
Koneksi Cabang	Berfungsi untuk menampilkan status koneksi dari server cabang
Koneksi puast	Berfungsi untuk menampilkan status koneksi dari server pusat.

4.1.9.3 Implementasi Antar Muka Sikopdit Server

Antar muka sikodpdit server adalah antar muka yang akan digunakan untuk mengakeses *Database*pusat. Berikut adalah penjelasannya:

Menu	Deskripsi
File	Berisi sub menu Keluar
Keluar	Berfungsi untuk keluar dari sistem
Pengaturan	Berisi menu koneksi
Koneksi	Berfungsi untuk mengatur koneksi *Database*
Tampil Data	Berisi sub menu Data Cabang, Data Anggota, Data Simpanan, Data Pinjaman dan Data Transaksi
Data Cabang	Berfungsi untuk menampilkan data seluruh cabang Rahastra CU
Data Anggota	Berfungsi untuk menampilkan data anggota dari seluruh cabang
Data Simpanan	Berisi sub menu Produk Simpanan, Saldo Simpanan Saham, Salso Simpanan Non Saham
Produk Simpanan	Berfungsi untuk menampilkan produk simpanan
Saldo Simpanan Sahan	Berfungsi untuk menampilkan data dari saldo simpanan saham
Saldo Simpanan Non Saham	Berfungsi untuk menampilkan data dari saldo simpanan non saham
Data Pinjaman	Berisi submenu Produk Pinjaman, Pinjaman Anggota, Saldo Pinjaman, Angsuran Pinjaman, Status Pinjaman
Produk	Berfungsi untuk menampilkan data dari produk

Pinjaman	pinjaman
Pinjaman Anggota	Berfungsi untuk menampilkan data dari pinjaman anggota
Sakdo Pinjaman	Berfungsi untuk menampilkan data dari saldo pinjaman
Angsuran Pinjaman	Berfungsi untuk menampilkan data dari angsuran pinjaman
Status Pinjaman	Berfungsi untuk menampilkan data dari status pinjaman
Data Transaksi	Berisi submenu Penarikan Simpanan Saham, Penarikan Simpanan Non Saham, Penarikan Saldo Pinjaman, Setoran Simpanan Saham, setoran Simpana Non Saham, Setoran Pinjaman
Penarikan Simpanan Saham	Berfungsi untuk menampilkan data dari penarikan simpanan saham
Penarikan Simpanan Non Saham	Berfungsi untuk menampilkan data dari penarikan simpanan non saham
Penarikan Saldo Pinjaman	Berfunsi untuk menampilkan data dari penarikan saldo pinjaman
Setoran Simpanan Saham	Nerfunsi unruk menampikan data dari setoran simpanan saham
Setoran Simpanan Non Saham	Berfungsi unruk menampilkan data dari setoran simpanan non saham
Setoran Pinjaman	Berfungsi untuk menampilkan data dari setoran pinjaman
Bantuan	Berisi submenu Tentang Kami
Tentang Kami	Berisi informasi mengenai pemmmmbuat aplikasi

4.2 Penujian

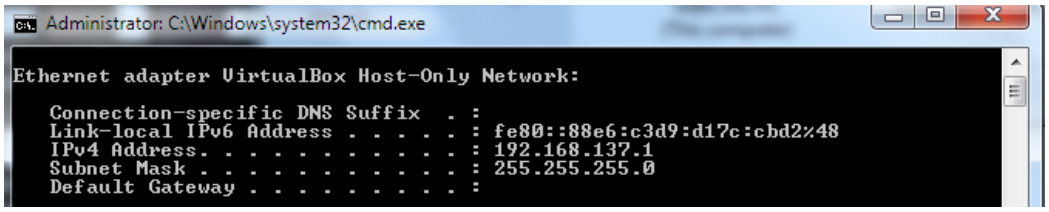
4.2.1 Pengujian Alpha

Pengujian alpha ialah pengujian sistem yang dibangun yang merepresentasikan kajian pokok dari spesifikasi, perangcangan dan pengkodean perangkat lunak. Pengujian ini menggunkan metode *black-box.* Pengujian *black-box* berfokus pada persyaratan fungsional perangkat lunak.

4.2.1.1 Pengujian Koneksi Jaringan VPN

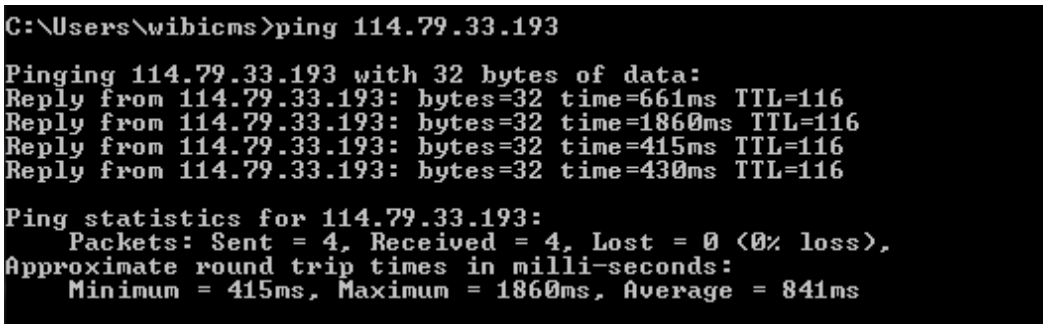
Pengujian koeneksi VPN dilakukan dengan melakukan *ping* dari komputer cabang ke komputer pusat.

1. Cek *IP Address* sebelum melakukan koneksi VPN



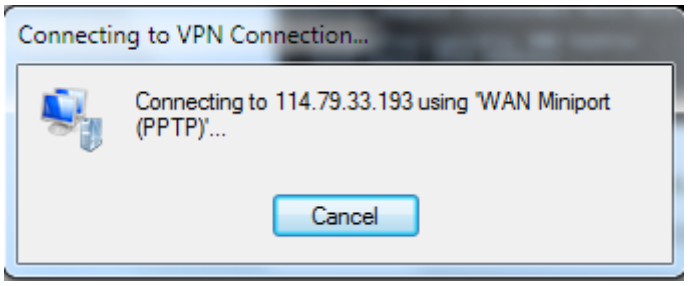
Gambar 4. 4 IP Address Sebelum VPN

2. Cek koneksi ke VPN Server



Gambar 4. 5 Cek Koneksi VPN Server

3. Lakukan koenksi ke VPN Server



Gambar 4. 6 Proses Koneksi VPN

4. Setelah koneksi berhasil, maka akan mendapatkan *IP Address* baru



Gambar 4. 7 IP Address Setelah Koneksi VPN

4.2.1.2 Pengujian Sinkronisasi *Database* Cabang Ke Pusat

Pengujian sinkroniasi *Database* cabang ke *Database* pusat dilakukan dengan melihat respon dari masing-masing tabel.

Tabel 4. 7 Hasil Pengujian Sinkroniasi Dari Cabang Ke Pusat

Nama Tabel	Jumlah Data Sinkronisasi	Keterangan
Cabang	614	Data berhasil disinkronisasi
User	3	Data berhasil disinkronisasi
Rekening	614	Data berhasil disinkronisasi

Anggota	614	Data berhasil disinkronisasi
Pembukaan	614	Data berhasil disinkronisasi
Penutupan	0	Tidak ada data sinkronisasi
Simpanan Saham	614	Data berhasil disinkronisasi
Setoran Saham	0	Tidak ada data sinkronisasi
Penarikan Saham	0	Tidak ada data sinkronisasi
Tabungan	0	Tidak ada data sinkronisasi
Tabungan Anggota	0	Tidak ada data sinkronisasi
Penarikan Tabungan	0	Tidak ada data sinkronisasi
Setoran Tabungan	0	Tidak ada data sinkronisasi
Pinjaman	0	Tidak ada data sinkronisasi
Pinjaman Anggota	0	Tidak ada data sinkronisasi
Angsuran	0	Tidak ada data sinkronisasi
Saldo Pinjaman	0	Tidak ada data sinkronisasi
Setoran Pinjaman	0	Tidak ada data sinkronisasi
Penarikan Saldo Pinjaman	0	Tidak ada data sinkronisasi
Status pinjaman	0	Tidak ada data sinkronisasi

4.2.1.3 Pengujian Sinkronisasi *Database* Pusat Ke Cabang

Pengujian sinkroniasi *Database* pusat ke *Database* cabang dilakukan dengan melihat respon dari masing-masing tabel.

Tabel 4. 8 Hasil Pengujian Sinkroniasi Dari Pusat Ke Cabang

Nama Tabel	Jumlah Data Sinkronisasi	Keterangan
Cabang	2	Data berhasil disinkronisasi
Simpanan Saham	4	Data berhasil disinkronisasi
Setoran Saham	10	Data berhasil disinkronisasi
Penarikan Saham	1	Data berhasil disinkronisasi
Tabungan	2	Data berhasil disinkronisasi
Tabungan Anggota	2	Data berhasil disinkronisasi
Penarikan Tabungan	2	Data berhasil disinkronisasi
Setoran Tabungan	2	Data berhasil disinkronisasi
Pinjaman	2	Data berhasil disinkronisasi
Pinjaman Anggota	1	Data berhasil disinkronisasi
Angsuran	12	Data berhasil disinkronisasi
Saldo Pinjaman	1	Data berhasil disinkronisasi
Setoran Pinjaman	1	Data berhasil disinkronisasi
Penarikan Saldo Pinjaman	1	Data berhasil disinkronisasi
Status pinjaman	1	Data berhasil disinkronisasi

4.2.1.4 Pengujian Replikasi *Database* Pusat

Tabel 4. 9 Hasil Pengujian Replikasi

Nama Tabel	Jumlah Record	
	Master	Slave
Cabang	614	614

User	3	3
Rekening	614	614
Anggota	614	614
Pembukaan	614	614
Penutupan	0	0
Simpanan Saham	614	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

4.2.1.5 Rencana Pengujian Aplikasi

Rencana pengujian akan dilakukan dengan menguji dalam 2 tahap yakni alpha dan beta. Pengujian perangkat lunak Sistem Informasi Koperasi Kredit Rahastara CU menggunakan data anggota yang ada di Rahastra CU.

Tabel 4. 10 Rencana Pengujian Aplikasi

Item Uji	Detail Pengujian	Jenis Pengujian
Lagin Admin	Verifikasi username	*Black box*
	Verifikasi *password*	*Black box*
	Verifikasi hak akses	*Black box*
Ubah Password	Edit Password	*Black box*
Sinkronisasi *Database*	Sinkronisai ke pusat	*Black box*
	Sinkronisasi dari pusat	*Black box*
Pengaturan	Pengaturan cabang	*Black box*
	Pengturan koneksi *Database* pusat	*Black box*
	Pengaturan koneksi *Database* cabang	*Black box*

1. Pengujian Login

Tabel 4. 11 Pengujian Login

Kasus dan Hasil Uji Data (Data Normal)	
Data masukan	*Username* : admin, *password* : admin, level: administrator
Yang diharapkan	Username, password dan level diterima
Pengamatan	*Username, password* dan level diterima dan masuk ke halaman administrator, sesuai dengan yang diharapkan
Kesimpulan	Diterima
Kasus dan Hasil Uji (Data Salah)	
Data masukan	*Username* : user, *password* : pass, level: administrator
Yang diharapkan	Kembali ke form login. Menampilkan pesan kesalahan login:"*Username atau*

	password salah".
Pengamatan	Menampilkan pesan login dan kembali ke form login. Sesuai dengan yang diharapkan.
Kesimpulan	Diterima

2. Pengujian Ubah *Password*

Tabel 4. 12 Pengujian Ubah *Password*

Kasus dan Hasil Uji Data (Data Normal)	
Data masukan	*Username* dan *password*
Yang diharapkan	Username dan password admin diubah dan disimpan kedalam sistem. Menampilkan pesan berhasil ubah di ubah.
Pengamatan	Username dan password baru di simpan
Kesimpulan	Diterima
Kasus dan Hasil Uji (Data Salah)	
Data masukan	*Username* dan *password*
Yang diharapkan	Menampilkan pesan kesalahan ubah password : "*file admin.txt tidak ditemukan"*. Kembali ke form ubah password.
Pengamatan	Username dan password gagal disimpan.
Kesimpulan	Diterima

3. Pengujian Sinkronisasi *Database*

Tabel 4. 13 Pengujian Sinkronisasi Ke Pusat

Kasus dan Hasil Uji Data (Data Normal)	
Data masukan	Waktu sinkronisasi
Yang diharapkan	Data yang ada di *Database* cabang akan tersinkronisasi ke *Database* pusat.
Pengamatan	Data cabang berhasil di sinkronisasikan ke *Database* pusat,
Kesimpulan	Diterima
Kasus dan Hasil Uji (Data Salah)	
Data masukan	Waktu sinkonisai
Yang diharapkan	Data tidak tersinkronisasi. Menampilkan pesan kesalahan : *"Sinkronisasi gagal".*
Pengamatan	Data tidak tersinkronisasi dan menampilkan pesan kesalahan
Kesimpulan	Diterima

Tabel 4. 14 Pengujian Sinkronisasi Dari Pusat

Kasus dan Hasil Uji Data (Data Normal)	
Data masukan	Waktu sinkronisasi
Yang diharapkan	Data yang ada di *Database* cabang akan tersinkronisasi ke *Database* pusat.
Pengamatan	Data cabang berhasil di sinkronisasikan ke *Database* pusat,
Kesimpulan	Diterima
Kasus dan Hasil Uji (Data Salah)	
Data masukan	Waktu sinkonisai
Yang diharapkan	Data tidak tersinkronisasi. Menampilkan pesan kesalahan : *"Sinkronisasi gagal".*
Pengamatan	Data tidak tersinkronisasi dan menampilkan pesan kesalahan
Kesimpulan	Diterima

4. Pengujian Pengaturan

A. Pengujian Pengaturan Cabang

Tabel 4. 15 Pengujian Edit cabang

Kasus dan Hasil Uji Data (Data Normal)	
Data masukan	Kode cabang, nama cabang, alamat cabang, kota
Yang diharapkan	Kode cabang, nama cabang, alamat cabang, kota di update. Menampilkan pesan: "*Data cabang berhasil di ubah".* Menampilkan data cabang didalam tabel
Pengamatan	Data cabang diubah dan ditampilkan dalam table dan menampilkan pesan sukses.
Kesimpulan	Diterima
Kasus dan Hasil Uji (Data Salah)	
Data masukan	Kode cabang, nama cabang, alamat cabang, kota
Yang diharapkan	Menampilkan pesan kesalahan ubah cabang: "*tidak dapat melakukan koneksi. Silakan periksa kembali konfigurasi koneksi".* Kembali ke form setup detail cabang.

Pengamatan	Kode cabang, nama cabang, alamat cabang, kota gagal di ubah.
Kesimpulan	Diterima

Tabel 4. 16 Pengujian Hapus cabang

Kasus dan Hasil Uji Data (Data Normal)	
Data masukan	Kode cabang, nama cabang, alamat cabang, kota
Yang diharapkan	Kode cabang, nama cabang, alamat cabang, kota di hapus. Menampilkan pesan: "*Data cabang berhasil di hapus".* Menampilkan data cabang didalam tabel
Pengamatan	Data cabang diubah dan ditampilkan dalam table dan menampilkan pesan sukses.
Kesimpulan	Diterima
Kasus dan Hasil Uji (Data Salah)	
Data masukan	Kode cabang, nama cabang, alamat cabang, kota
Yang diharapkan	Menampilkan pesan kesalahan hapus cabang: "*tidak dapat melakukan koneksi. Silakan periksa kembali konfigurasi koneksi".* Kembali ke form setup detail cabang.
Pengamatan	Kode cabang, nama cabang, alamat cabang, kota gagal di hapus.
Kesimpulan	Diterima

Tabel 4. 17 Pengujian Simpan cabang

Kasus dan Hasil Uji Data (Data Normal)	
Data masukan	Kode cabang, nama cabang, alamat cabang, kota
Yang diharapkan	Kode cabang, nama cabang, alamat cabang, kota di simpan. Menampilkan pesan: "*Data cabang berhasil di simpan".* Menampilkan data cabang didalam tabel
Pengamatan	Data cabang disimpan dan ditampilkan dalam table dan menampilkan pesan sukses.
Kesimpulan	Diterima
Kasus dan Hasil Uji (Data Salah)	
Data masukan	Kode cabang, nama cabang, alamat cabang, kota
Yang diharapkan	Menampilkan pesan kesalahan simpan cabang: "*tidak dapat melakukan koneksi. Silakan periksa kembali konfigurasi koneksi".*

	Kembali ke form setup detail cabang.
Pengamatan	Kode cabang, nama cabang, alamat cabang, kota gagal di simpan.
Kesimpulan	Diterima

B. Pengujian Koneksi *Database* Pusat

Tabel 4. 18 Pengujian Tes Koneksi *Database* Utama Pusat

Kasus dan Hasil Uji Data (Data Normal)	
Data masukan	Host, user, password, *Database*
Yang diharapkan	Host, user, password, *Database* dapat terkoneksi. Menampilkan pesan: "*Koneksi berhasil".*
Pengamatan	*Database* utama terkoneksi dan menampilkan pesan sukses.
Kesimpulan	Diterima
Kasus dan Hasil Uji (Data Salah)	
Data masukan	Host, user, password, *Database*
Yang diharapkan	Menampilkan pesan kesalahan gagal koneksi: "*tidak dapat melakukan koneksi. Silakan periksa kembali konfigurasi koneksi".* Kembali ke form pengaturan koneksi server *Database* utama pusat.
Pengamatan	Host, user, password, *Database* gagal terkoneksi.
Kesimpulan	Diterima

Tabel 4. 19 Pengujian Simpan Koneksi *Database* Utama Pusat

Kasus dan Hasil Uji Data (Data Normal)	
Data masukan	Host, user, password, *Database*
Yang diharapkan	Host, user, password, *Database* di simpan. Menampilkan pesan: "*Konfigurasi berhasil di simpan".*
Pengamatan	*Database* utama tersimpan dan menampilkan pesan sukses.
Kesimpulan	Diterima
Kasus dan Hasil Uji (Data Salah)	
Data masukan	Host, user, password, *Database*

Yang diharapkan	Menampilkan pesan kesalahan gagal simpan: "*tidak dapat melakukan koneksi. Silakan periksa kembali konfigurasi koneksi".* Kembali ke form pengaturan koneksi server *Database* utama pusat.
Pengamatan	Host, user, password, *Database* gagal tersimpan.
Kesimpulan	Diterima

Tabel 4. 20 Pengujian Test Koneksi *Database* Temp Pusat

Kasus dan Hasil Uji Data (Data Normal)	
Data masukan	Host, user, password, *Database*
Yang diharapkan	Host, user, password, *Database* dapat terkoneksi. Menampilkan pesan: "*Koneksi berhasil".*
Pengamatan	*Database* utama terkoneksi dan menampilkan pesan sukses.
Kesimpulan	Diterima
Kasus dan Hasil Uji (Data Salah)	
Data masukan	Host, user, password, *Database*
Yang diharapkan	Menampilkan pesan kesalahan gagal koneksi: "*tidak dapat melakukan koneksi. Silakan periksa kembali konfigurasi koneksi".* Kembali ke form pengaturan koneksi server *Database* Temp.
Pengamatan	Host, user, password, *Database* gagal terkoneksi.
Kesimpulan	Diterima

Tabel 4. 21 Pengujian Simpan Koneksi *Database* Temp Pusat

Kasus dan Hasil Uji Data (Data Normal)	
Data masukan	Host, user, password, *Database*
Yang diharapkan	Host, user, password, *Database* di simpan. Menampilkan pesan: "*Konfigurasi berhasil di simpan".*
Pengamatan	*Database* utama tersimpan dan menampilkan pesan sukses.
Kesimpulan	Diterima
Kasus dan Hasil Uji (Data Salah)	
Data masukan	Host, user, password, *Database*
Yang diharapkan	Menampilkan pesan kesalahan gagal simpan: "*tidak dapat melakukan koneksi. Silakan periksa kembali*

	konfigurasi koneksi". Kembali ke form pengaturan koneksi server *Database* utama pusat.
Pengamatan	Host, user, password, *Database* gagal tersimpan.
Kesimpulan	Diterima

C. Pengujian Koneksi *Database* Utama Cabang

Tabel 4. 22 Pengujian Tes Koneksi *Database* Cabang

Kasus dan Hasil Uji Data (Data Normal)	
Data masukan	Host, user, password, *Database*
Yang diharapkan	Host, user, password, *Database* dapat terkoneksi. Menampilkan pesan: "*Koneksi berhasil".*
Pengamatan	*Database* utama terkoneksi dan menampilkan pesan sukses.
Kesimpulan	Diterima
Kasus dan Hasil Uji (Data Salah)	
Data masukan	Host, user, password, *Database*
Yang diharapkan	Menampilkan pesan kesalahan gagal koneksi: "*tidak dapat melakukan koneksi. Silakan periksa kembali konfigurasi koneksi".* Kembali ke form pengaturan koneksi server *Database* utama cabang.
Pengamatan	Host, user, password, *Database* gagal terkoneksi.
Kesimpulan	Diterima

Tabel 4. 23 Pengujian Simpan Koneksi *Database* Utama Cabang

Kasus dan Hasil Uji Data (Data Normal)	
Data masukan	Host, user, password, *Database*
Yang diharapkan	Host, user, password, *Database* di simpan. Menampilkan pesan: "*Konfigurasi berhasil di simpan".*
Pengamatan	*Database* utama tersimpan dan menampilkan pesan sukses.
Kesimpulan	Diterima
Kasus dan Hasil Uji (Data Salah)	
Data masukan	Host, user, password, *Database*
Yang diharapkan	Menampilkan pesan kesalahan gagal simpan: "*tidak*

	dapat melakukan koneksi. Silakan periksa kembali konfigurasi koneksi". Kembali ke form pengaturan koneksi server *Database* utama cabang.
Pengamatan	Host, user, password, *Database* gagal tersimpan.
Kesimpulan	Diterima

Tabel 4. 24 Pengujian Test Koneksi *Database* Temp Cabang

Kasus dan Hasil Uji Data (Data Normal)	
Data masukan	Host, user, password, *Database*
Yang diharapkan	Host, user, password, *Database* dapat terkoneksi. Menampilkan pesan: "*Koneksi berhasil".*
Pengamatan	*Database* utama terkoneksi dan menampilkan pesan sukses.
Kesimpulan	Diterima
Kasus dan Hasil Uji (Data Salah)	
Data masukan	Host, user, password, *Database*
Yang diharapkan	Menampilkan pesan kesalahan gagal koneksi: "*tidak dapat melakukan koneksi. Silakan periksa kembali konfigurasi koneksi".* Kembali ke form pengaturan koneksi server *Database* Temp cabang.
Pengamatan	Host, user, password, *Database* gagal terkoneksi.
Kesimpulan	Diterima

Tabel 4. 25 Pengujian Simpan Koneksi *Database* Temp Cabang

Kasus dan Hasil Uji Data (Data Normal)	
Data masukan	Host, user, password, *Database*
Yang diharapkan	Host, user, password, *Database* di simpan. Menampilkan pesan: "*Konfigurasi berhasil di simpan".*
Pengamatan	*Database* utama tersimpan dan menampilkan pesan sukses.
Kesimpulan	Diterima
Kasus dan Hasil Uji (Data Salah)	
Data masukan	Host, user, password, *Database*
Yang diharapkan	Menampilkan pesan kesalahan gagal simpan: "*tidak dapat melakukan koneksi. Silakan periksa kembali konfigurasi koneksi".*

	Kembali ke form pengaturan koneksi server *Database* utama pusat.
Pengamatan	Host, user, password, *Database* gagal tersimpan.
Kesimpulan	Diterima

4.2.1.6 Hasil Pengujian Aplha

Berdasarkan hasil pengujian alpha (fungsional) dengan kasus uji diatas dapat ditarik kesimpulan sebagai berikut:

1. Jika data yang dimasukan benar maka sistem secara fungsional akan mengeluarkan keluaran sesuai dengan harapan.
2. Jika data yang dimasukan salah maka sistem akan mengeluarkan pesan kesalahan dan data keluaran tidak akan sesuai dengan harapan.

Dengan kata lain bahwa pembangunan Sistem ini bebas kesalahan dan secara fungsional mengeluarkan hasil yang sesuai dengan yang diharapkan.

4.2.2 Pengujian Beta

Pengujian beta dilakukan untuk dengan cara melakukan wawanacara dengan administrator yang ada di Rahastra CU. Berikut ini adalah daftar pertanyaan yang diajukan ketika wawancara?

1. Apakah sistem ini dapat melakukan sinkronisasi *database* dari setiap kantor cabang?
2. Apakah sistem ini mampu menjaga *database* tetap valid?

3. Apakah sistem ini dapat menyelesaikan permasalahan yang ada di Rahastra CU khusunya yang berkitan dengan masalah integritas *database*?

Kesimpulan Hasil Pengujian Beta

Berdasarkan hasil wawancara, maka berikut adalah jawaban dari setiap pertanyaan yang diajukan.

1. Sistem ini sudah mampu melakukan sinkronisasi *database* dari setiap cabang. Namun kedepannya diharapkan agar sinkronisasi dapat dilakukan secara *real time.*
2. Dengan metode sinktoniasi yang ada dalam sistem ini maka data dari setiap *database* kantor cabang tetap valid.
3. Masalah integritas *database* di Rahastra CU adalah masalah utana yang dihadapi oleh Rahastra CU dan sistem ini mampu menyelesaikan masalah tersebut.

Berdasarkan jawaban diatas maka administrator Rahastra CU cukup puas akan kinerja dari sistem. Maka dapat disimpulkan jika sistem yang dibuat telah sesuai dengan kriteria yang diinginkan.