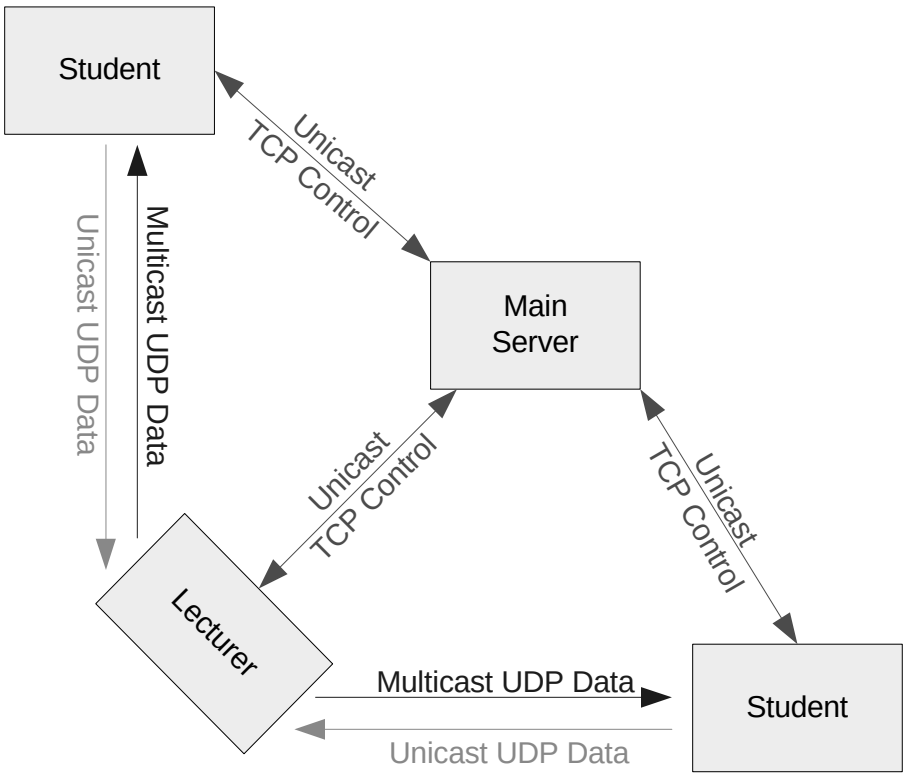
BAB 3

PERANCANGAN SISTEM

3.1 Blok Diagram Sistem

Perancangan aplikasi meliputi *server*, pengajar dan mahasiswa. *Server* akan melakukan koneksi dengan pengajar dan mahasiswa secara *unicast* dengan menggunakan protokol TCP. *Server* akan mengatur hubungan antara pengajar dan mahasiswa, seperti mengirimkan alamat IP *multicast* serta *port* yang akan digunakan, memberikan *list user* yang aktif, dan lain sebagainya. Aplikasi pengajar dan mahasiswa akan terkoneksi dengan *server* secara *unicast* dengan protokol TCP. pengajar mengirimkan data berupa video dan suara serta teks secara *multicast* ke mahasiswa, pengajar menerima video dan suara dari mahasiswa dengan menggunakan koneksi secara *unicast* dengan protokol UDP, serta teks dari mahasiswa menggunakan koneksi secara *multicast*. Aplikasi pengajar yang dibuat dapat menerima video dan suara dari mahasiswa maksimum satu mahasiswa dalam satu waktu. Video c*onference* dilakukan secara bergantian dari satu mahasiswa ke mahasiswa lain.

Pemakaian alamat IP untuk koneksi antara *server* dengan aplikasi mahasiswa dan pengajar menggunakan alamat IP komputer tempat aplikasi tersebut berjalan, sedangkan untuk pemakaian alamat IP *multicast* menggunakan alamat IP diantara 224.1.1.1 sampai dengan 224.1.1.254. Penggunaan *port unicast* dan *multicast* menggunakan nomor *port* diantara 1024 sampai dengan 65535. Penjelasan diatas dapat dilihat dari gambar 3.1 dibawah ini.

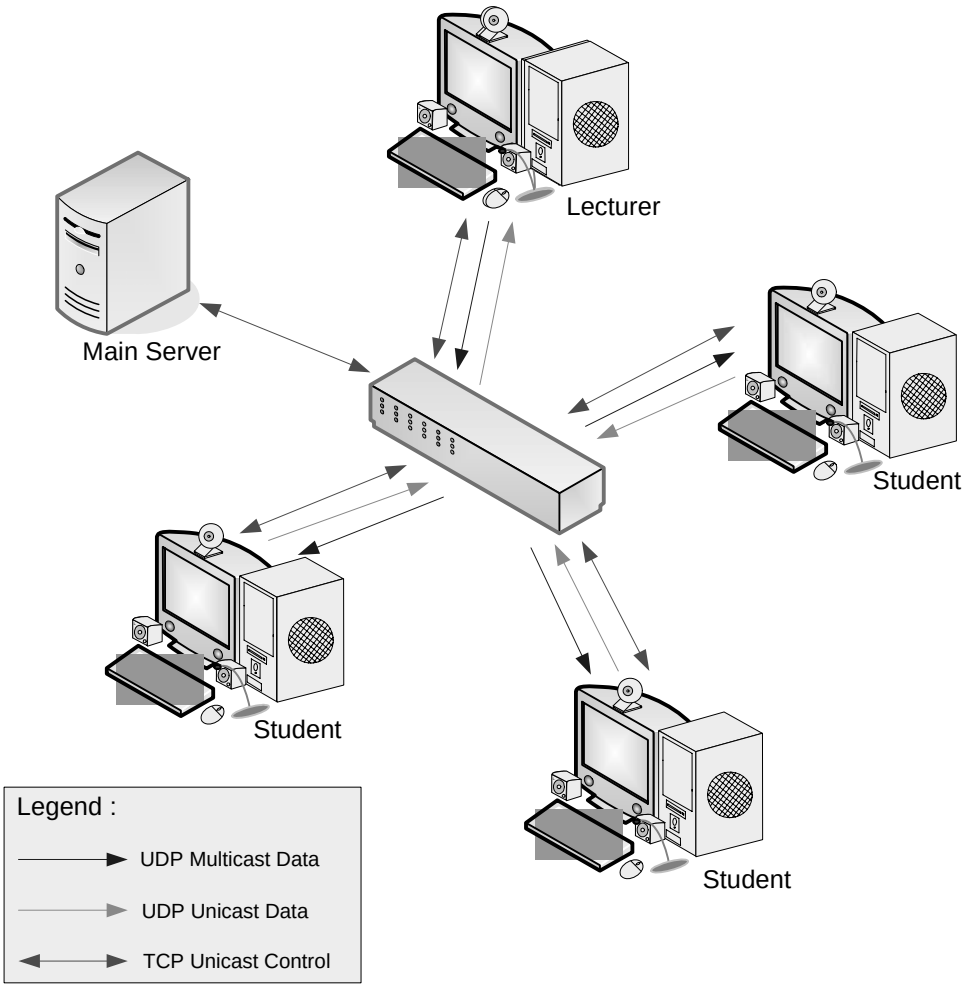


Gambar 3.1 Blok diagram sistem

3.2 Rancangan Sistem

Perangkat keras dari sistem aplikasi video *conference* antara lain komputer, *switch* atau *router*, *webcam*, *speaker* dan mikrofon. Komputer digunakan untuk menjalankan aplikasi yang dibuat. *Switch* atau *router* digunakan untuk menghubungkan komputer dalam jaringan. *Webcam* digunakan untuk mengirimkan video, *webcam* yang digunakan adalah *webcam* dengan koneksi USB (*Universal Serial Bus*). *Speaker* digunakan untuk mengeluarkan suara dari komputer, serta mikrofon digunakan untuk mengirimkan suara ke komputer. *Speaker* dan mikrofon yang digunakan menggunakan koneksi kabel *jack* 3,5 mm. Komputer-komputer pada sistem akan terhubung dengan mikrofon, *speaker* dan *webcam*, dan akan menjalankan aplikasi sebagai *server*, pengajar, dan mahasiswa. Komputer-komputer tersebut akan dihubungkan dalam jaringan dengan menggunakan perangkat *switch*

ataupun *router*. Perancangan perangkat keras yang digunakan dapat terlihat dari gambar 3.2 dibawah ini



Gambar 3.2 Perangkat keras sistem

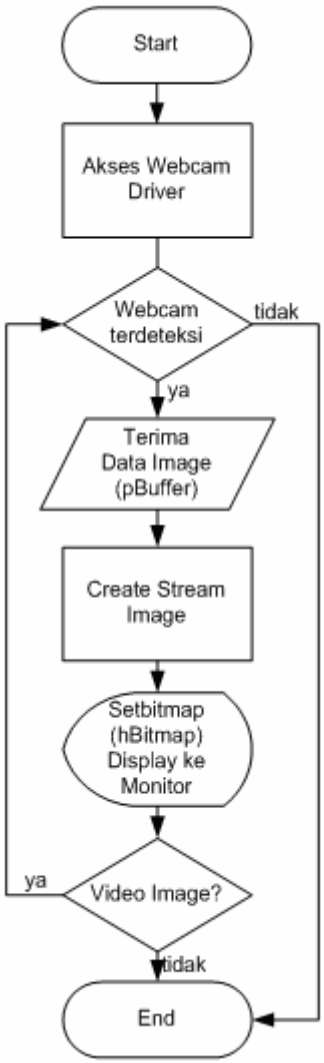
3.3 Diagram-diagram Alir Rutin Pendukung

Pada aplikasi pengajar dan mahasiswa dibutuhkan beberapa modul tambahan yang digunakan untuk melakukan teks, suara dan video *conference*. Modul tambahan ini bertujuan agar pembuatan aplikasi lebih mudah. Modul ini akan digunakan oleh aplikasi pengajar dan mahasiswa. Beberapa modul yang dibuat antara lain adalah modul akses w*ebcam*, modul akses jaringan UDP, modul pengiriman gambar, modul

penerimaan gambar, modul pengiriman suara, modul penerimaan suara, dan modul teks *chat*. Berikut adalah cara kerja dan fungsi dari modul-modul diatas.

3.3.1 Modul akses w*ebcam*

Modul ini berfungsi untuk melakukan pengaksesan *webcam* sehingga dapat menampilkan gambar pada aplikasi. Modul ini mengatur besar ukuran gambar yang akan ditampilkan ke layar *monitor*. Gambar yang ada ditampilkan pada aplikasi dalam format data BITMAP. *Webcam* akan memberikan data video yang dikirimkan secara terus menerus pada aplikasi. Modul akses *webcam* membuat agar data video yang diterima oleh aplikasi dari *webcam* merupakan rentetan *frame* gambar (video) yang akan membuat tampilan seperti gambar yang bergerak. Proses *update* gambar ke layar *monitor* tergantung dari kesiapan modul dalam mengambil data gambar. Bila data yang diterima telah ditampung dengan benar maka akan ditampilkan ke layar *monitor*. Berikut diagram alir untuk Modul Akses *Webcam*



Gambar 3.3 Diagram alir modul akses *webcam*

3.3.2 Modul akses jaringan UDP

Modul ini bertujuan agar aplikasi dapat mengakses jaringan yang ada dengan menggunakan alamat IP dan *Port* dengan protocol UDP. Modul ini digunakan agar antara aplikasi pengajar dan aplikasi mahasiswa dapat terjadi video *conference*. Penggunaan protocol UDP bertujuan agar pengiriman data yang dilakukan berlangsung secara cepat karena tidak melakukan *error correction*, sehingga akan didapatkan pengiriman data yang mendekati *real-time*. Adanya paket data yang

rusak dalam pengiriman dengan menggunakan protokol UDP ini dapat diabaikan karena pengiriman paket data dilakukan secara terus menerus, sehingga adanya satu atau dua paket data yang hilang atau rusak tidak akan menjadi masalah dalam pengiriman data tersebut. Modul ini terdiri atas dua bagian, yaitu:

- Akses jaringan UDP

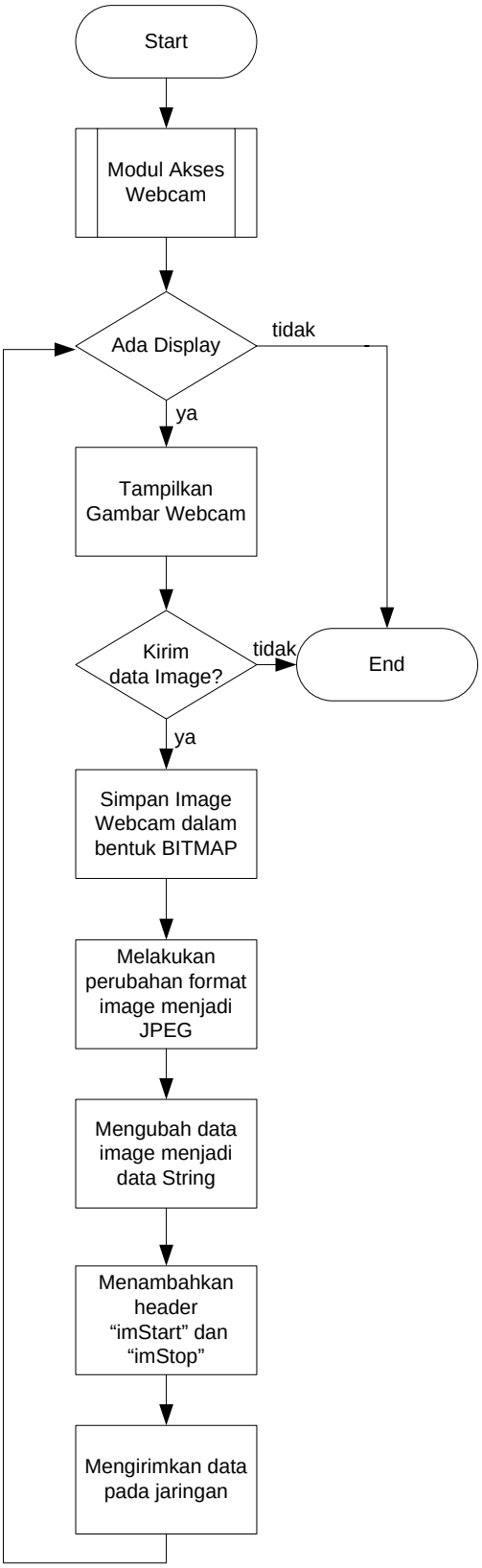
  Bagian ini bertugas untuk melakukan pengiriman dan penerimaan data dalam jaringan dengan menggunakan protokol UDP (*User Datagram Protocol*). Akses jaringan ini digunakan oleh aplikasi mahasiswa untuk melakukan pengiriman video dan suara kepada aplikasi pengajar. Perancangan sistem video *conference* membatasi hanya maksimum satu *user* aplikasi mahasiswa yang melakukan video *conference* dengan aplikasi pengajar, sehingga aplikasi pengajar tidak mendapat beban penerimaan data yang besar, dengan adanya pembatasan ini jalur data yang digunakan untuk melakukan *conference* tidak terlalu besar.

- Akses jaringan *multicast*

  Bagian ini bertugas untuk melakukan pengiriman data secara *multicast*, serta melakukan *join* ke dalam suatu grup *multicast* yang ada untuk menerima data yang ada dalam grup tersebut. Akses jaringan ini digunakan oleh aplikasi pengajar untuk mengirimkan data video dan suara, sehingga jumlah aplikasi mahasiswa yang banyak tidak akan mempengaruhi besar *bandwidth* yang digunakan dalam pengiriman data. Modul ini juga digunakan oleh aplikasi mahasiswa agar dapat melakukan *join* ke dalam suatu grup *multicast* dan menerima data dari pengajar.

3.3.3 Modul pengiriman gambar

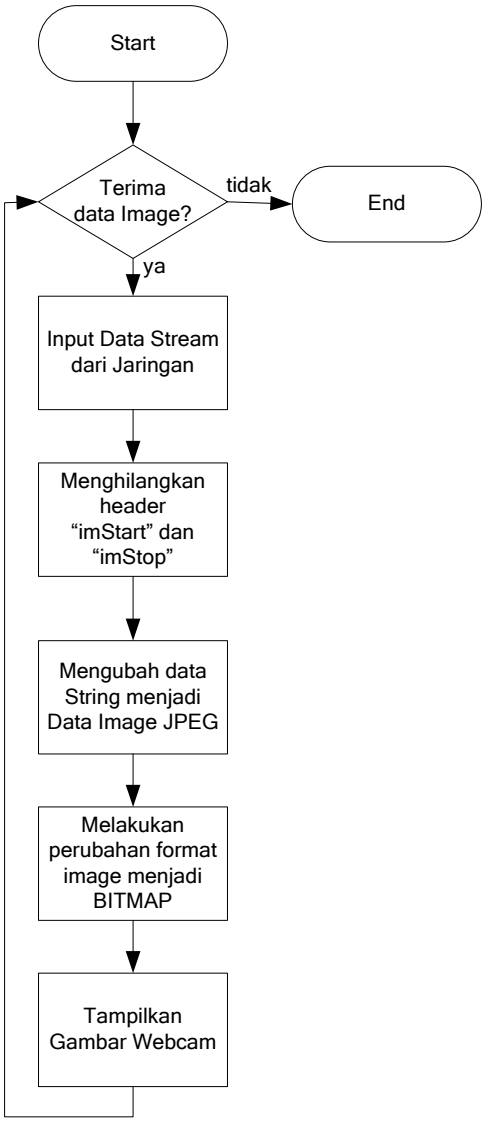
Modul ini bertugas untuk melakukan pengiriman data-data gambar dari aplikasi ke dalam jaringan komputer lokal. Modul ini bertugas untuk mengubah data gambar yang diterima dari *webcam*. Modul ini mengambil data gambar dalam format BITMAP kemudian melakukan kompresi menjadi bentuk data gambar dalam format. Data yang telah diubah dalam bentuk format JPEG siap dikirimkan dalam jaringan. Tujuan penggunaan format JPEG adalah format JPEG memliki kompresi data gambar yang tinggi sehingga besar ukuran data gambar akan semakin kecil dibanding sebelum kompresi. Perbandingan data gambar dengan resolusi 320x240 jika dengan format BITMAP akan mempunyai ukuran gambar 230.400 b*ytes* dan jika dengan format gambar JPEG akan memiliki ukuran gambar 18.000 -22.000 *bytes*. Pengiriman data yang dilakukan didalam jaringan merupakan data-data gambar yang diubah menjadi bentuk data berupa *string* yang mempunyai *header* yang unik. Awal *frame* diawali dengan ”imStart” dan akhir *frame* ditandai oleh ”imStop”. Data gambar yang telah diubah dalam bentuk *string* terdapat di antara dua batasan tersebut. Tujuan pemberian *header* yang unik agar data yang dikirimkan merupakan data gambar yang benar-benar berasal dari satu gambar *frame*. Berikut diagram alir untuk modul pengiriman gambar:



Gambar 3.4 Diagram alir modul pengiriman gambar

3.3.4 Modul penerimaan gambar

Modul ini bertugas untuk mengambil data-data yang dikirimkan oleh aplikasi melalui jaringan. Data-data yang diterima berupa *string* yang mempunyai *header* yang unik. Data *string* yang dikirimkan melalui jaringan ditandai dengan bagian yang menentukan awal dan akhir *frame string*. Awal *frame* diawali dengan ”imStart” dan akhir *frame* ditandai oleh ”imStop”. Data gambar dalam bentuk *string* terdapat di antara dua batasan tersebut. Data gambar dalam bentuk *string* yang diterima akan diubah kembali menjadi bentuk gambar dengan format JPEG. Modul akan mengubah dari format JPEG menjadi BITMAP agar dapat ditampilkan pada aplikasi. Gambar dalam format JPEG akan di-*uncompress* menjadi bentuk BITMAP, dimana perubahan ini bertujuan untuk menampilkan gambar pada layar *monitor* komputer. Aplikasi akan menampilkan gambar dalam bentuk rentetan data format BITMAP sehingga membentuk suatu gambar bergerak atau video pada aplikasi. Berikut diagram alir untuk modul penerimaan gambar.



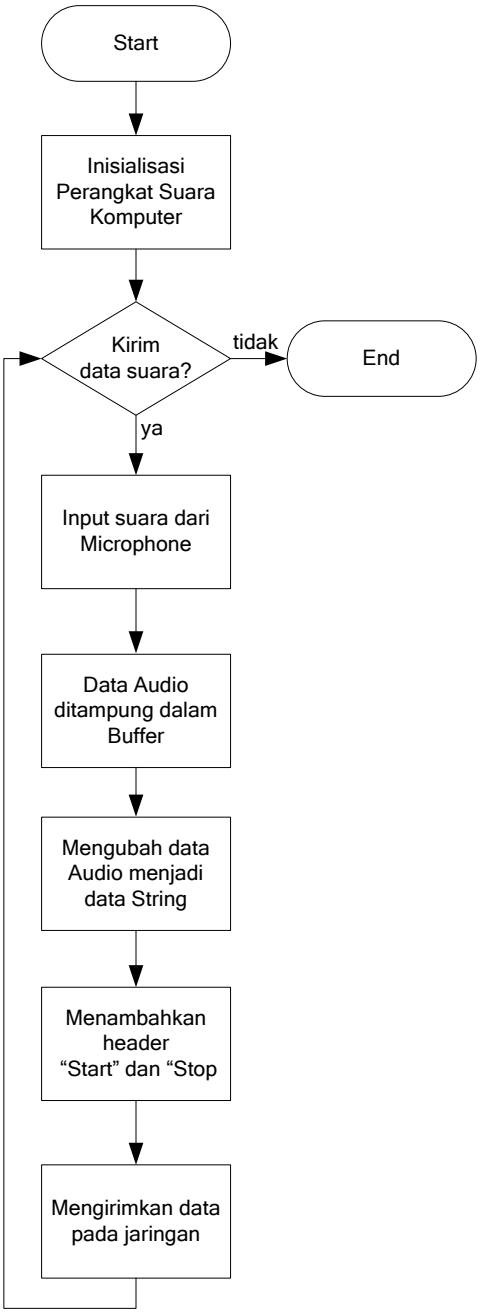
Gambar 3.5 Diagram alir modul penerimaan gambar

3.3.5 Modul pengiriman suara

Modul ini bertugas untuk mengirimkan data-data suara dari aplikasi ke dalam jaringan komputer lokal. Modul ini akan mengambil *input* berupa data suara yang berasal dari mikrofon. Modul ini mengatur besar ukuran suara yang akan digunakan serta menyediakan *memory buffer* untuk pengiriman suara dalam jaringan. Modul ini menggunakan format suara WAV dengan besar resolusi 8 *bit*,

serta frekuensi *sampling* 8 KHz, dengan *channel* suara *mono*. Aplikasi digunakan untuk video *conference*, dimana data yang dikirimkan adalah suara manusia, format suara tersebut telah mencukupi untuk melakukan percakapan dimana suara manusia berada dalam rentang 3 KHz - 4 KHz, karena itu tidak dibutuhkan resolusi data yang tinggi serta frekuensi *sampling* 8 KHz sudah cukup untuk melakukan *sampling* data suara manusia, penggunaan channel suara *mono* karena mikrofon yang digunakan mikrofon *mono*. Penggunaan format suara tersebut membuat besar pengiriman data suara mempunyai besar *bandwidth* sekitar 8 KBps untuk masing-masing *user*. Penggunaan format suara ini akan memperkecil data yang dikirimkan.

Pengiriman data suara dirancang menggunakan ukuran *buffer* dengan durasi *frame* sebesar 50 ms. Data-data suara yang akan dikirimkan dalam jaringan diubah menjadi data-data dalam bentuk *string*, dimana data *string* yang dikirimkan ditandai dengan bagian yang menentukan awal dan akhir *frame string*. Awal *frame* diawali dengan ”Start” dan akhir *frame* ditandai oleh ”Stop”. Data suara dalam bentuk *string* terdapat di antara dua batasan tersebut. Tujuan pemberian *header* yang unik agar data yang dikirimkan merupakan data suara yang benar-benar berasal dari satu durasi *frame*. Berikut diagram alir untuk modul pengiriman suara

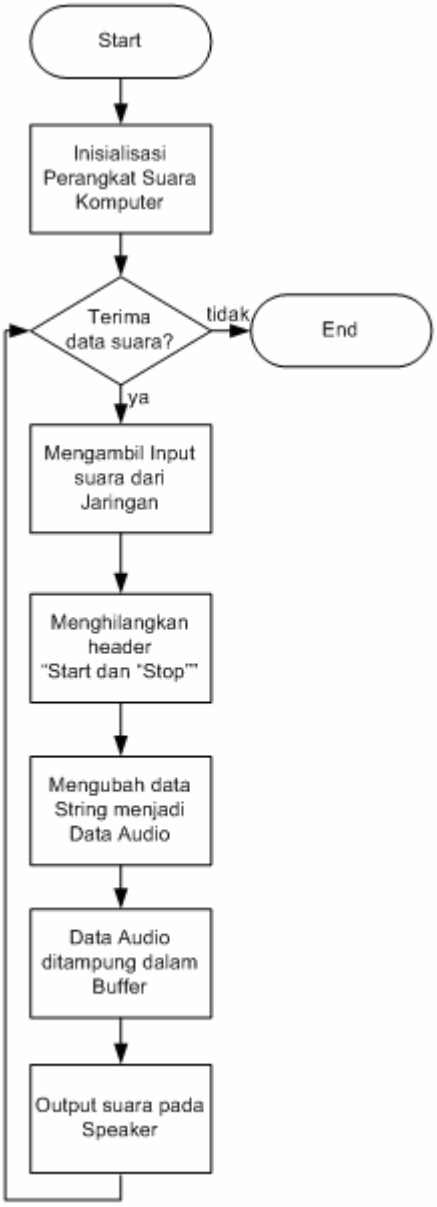


Gambar 3.6 Diagram alir modul pengiriman suara

3.3.6 Modul penerimaan suara

Modul ini bertugas untuk mengambil data-data suara dari aplikasi dalam jaringan yang ada. Modul ini akan mengeluarkan *output* berupa data suara ke

dalam perangkat *speaker*. Modul ini mengatur besar ukuran suara yang akan digunakan serta menyediakan *memory buffer* untuk penerimaan suara dalam jaringan. Modul ini menggunakan format suara WAV yang sama seperti pengiriman suara dimana besar resolusi data sebesar 8 bit, frekuensi *sampling* 8Khz, serta *channel* suara *mono*. Penerimaan data suara harus memiliki format yang sama dengan pengiriman suara agar data suara dapat terdengar dengan baik. Durasi data yang dikirimkan harus sama antara pengirim dan penerima, sehingga digunakan ukuran *buffer* yang sama dengan pengirim yaitu dengan durasi *frame* sebesar 50 ms. Data-data suara yang diterima dalam jaringan didapatkan berupa data-data dalam bentuk *string*, dimana data *string* yang dikirimkan ditandai dengan bagian yang menentukan awal dan akhir *frame string*. Awal *frame* diawali dengan ”Start” dan akhir *frame* ditandai oleh ”Stop”. Data suara dalam bentuk *string* terdapat di antara dua batasan tersebut. Data suara tersebut kemudian diubah menjadi suara yang kemudian dikeluarkan dalam bentuk *output* suara dalam *speaker*. Berikut diagram alir untuk modul penerimaan suara

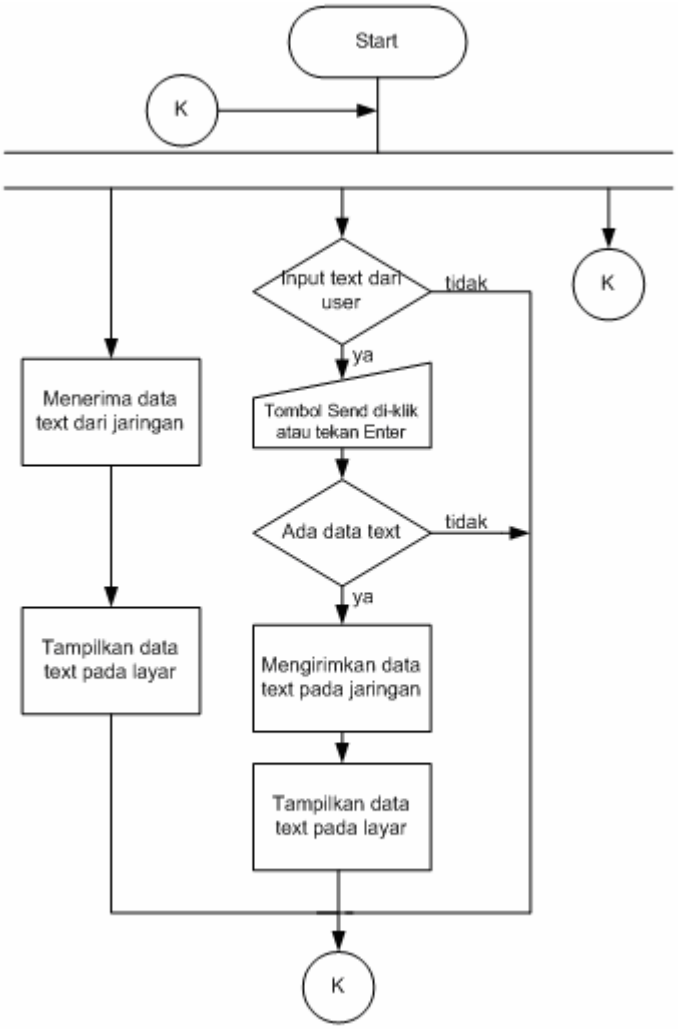


Gambar 3.7 Diagram alir modul penerimaan suara

3.3.7 Modul teks *chat*

Modul ini digunakan untuk melakukan teks *chat*. Modul ini bertugas untuk melakukan pengiriman dan penerimaan data berupa teks. Modul ini mempunyai bagian pengiriman dan penerimaan data teks. Bagian pengiriman bertugas untuk

menampilkan teks pada aplikasi dan mengirimkan data teks melalui jaringan dalam bentuk data *string*. Bagian penerimaan bertugas menerima data teks yang diterima dalam jaringan kemudian ditampilkan pada aplikasi. Berikut diagram alir untuk modul teks *chat*



Gambar 3.8 Diagram alir modul teks *chat*

3.4 Modul-modul Sistem dan Cara Kerjanya

Sistem yang dibuat terbagi menjadi tiga aplikasi yaitu *server*, pengajar dan mahasiswa. Aplikasi yang dibuat menggunakan *header* data tertentu yang digunakan

untuk berkomunikasi. Pada saat pengiriman data dari *server* ke pengajar atau mahasiswa dan sebaliknya akan dikirimkan *header* data yang berfungsi sebagai penanda perintah yang terdiri dari enam huruf.

3.4.1 Aplikasi *Server*

Aplikasi s*erver* berfungsi untuk menghubungkan antara aplikasi pengajar dan aplikasi mahasiswa. Penggunaan *server* pada sistem akan membuat pengajar dan mahasiswa tidak perlu melakukan banyak konfigurasi. *Server* akan melakukan pengaturan pada sisi mahasiswa dan pengajar secara otomatis, dengan hal tersebut maka *user* tidak akan disulitkan dalam penggunaan aplikasi dan mencegah terjadinya kesalahan konfigurasi. Aplikasi pengajar dan mahasiswa akan terkoneksi dengan *server* secara *unicast* dengan protokol TCP (*Transmission Control Protocol*) dengan tujuan agar terjadinya pengiriman data yang terjamin tanpa ada kesalahan. Protokol TCP menggunakan fasilitas *error-correction* yang akan memperbaiki paket data yang rusak dalam pengiriman sehingga dipastikan bahwa paket data yang dikirimkan akan diterima dengan baik oleh penerima. Aplikasi *server* mengirimkan data-data perintah untuk berkomunikasi dengan pengajar dan mahasiswa, perintah yang digunakan dapat dilihat pada tabel 3.1 dibawah ini.

Tabel 3.1 Data perintah pada aplikasi s*erver* (server ke pengajar atau mahasiswa)

Protokol	Keterangan
"USREXT"	Aplikasi pengajar atau mahasiswa yang baru aktif dan terkoneksi dengan aplikasi *server* akan mengirimkan status berupa

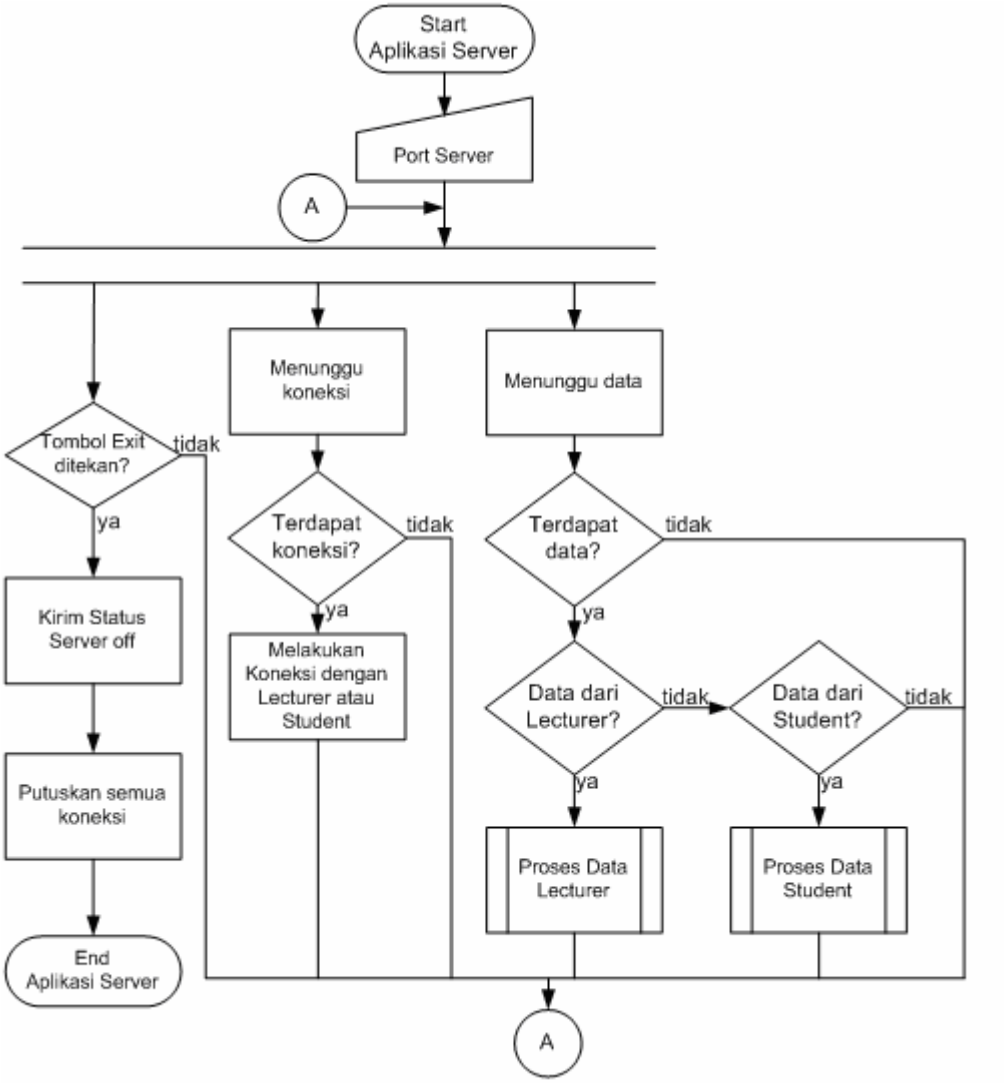
	nickname ke *server*. *Nickname* setiap *user* baik pengajar dan mahasiswa tidak boleh sama, apabila terdapat *nickname* yang sama maka *server* akan mengirimkan protokol ini. Protokol ini akan membuat *user* harus mengubah *nickname* mereka hingga berbeda dari *user* yang telah ada, jika *nickname* yang digunakan sudah unik atau berbeda maka *server* akan menyimpan informasi *nickname* tersebut ke dalam *list*. Pengajar akan terdapat pada *list* pengajar, dan mahasiswa akan terdapat pada *list* mahasiswa. "USREXT" & Chr$(2) & Chr$(4)
"SERIPM"	Protokol ini akan dikirimkan kepada pengajar apabila *nickname* mereka telah unik, saat *nickname* pengajar telah unik maka *server* akan memilih alamat IP *multicast* serta *port* yang akan digunakan oleh pengajar untuk mengirimkan video dan suara serta teks. *Server* akan mengirimkan alamat IP *multicast* dan *port* kepada pengajar, dan jika mahasiswa yang terkoneksi memilih pengajar tersebut maka akan diberikan perintah ini dan juga alamat IP *multicast* dan *port* yang sama dengan pengajar sehingga pengajar dan mahasiswa tergabung dalam satu grup *multicast*. "SERIPM" & Chr$(2) & ipmulti & vbcrlf & ipmultiport & Chr$(4)
"LSTSER"	Protokol ini dikirimkan saat aplikasi mahasiswa telah memiliki *nickname* yang unik, protokol ini akan dikirimkan dengan *list*

	pengajar yang saat itu terkoneksi. *List-list* pengajar tersebut akan dipilih oleh mahasiswa. Aplikasi mahasiswa yang telah memilih pengajar akan diberikan alamat IP *multicast* dan *port* dari pengajar sehingga ia akan tergabung dalam grup *multicast* yang sama dengan pengajar. "LSTSER" & Chr$(2) & server1 & vbCrLf & server2 & vbCrLf & server3 & Chr$(4)
“LSTUSR”	Protokol ini dikirimkan saat terdapat mahasiswa baru yang terkoneksi atau terdapat mahasiswa yang keluar dari video *conference*. Aplikasi *Server* akan mengirimkan protokol beserta *list user* baru yang tergabung dalam satu grup *multicast* video *conference* kepada aplikasi pengajar dan aplikasi mahasiswa. Aplikasi pengajar dan mahasiswa akan melakukan *update list user*. "LSTUSR" & Chr$(2) & User1 & vbCrLf & User2 & vbCrLf & User3 & Chr$(4)
“SEROUT”	Protokol ini akan dikirimkan ke mahasiwa apabila pengajar terputus atau telah menghentikan video *conference* dan keluar dari *server*. Aplikasi *Server* akan mengirimkan protokol ini kepada semua aplikasi mahasiswa yang tergabung dalam grup video *conference* yang sama dengan aplikasi pengajar. Aplikasi mahasiswa akan memutuskan koneksi ke *server* saat mendapatkan perintah ini, saat itu maka aplikasi mahasiswa akan

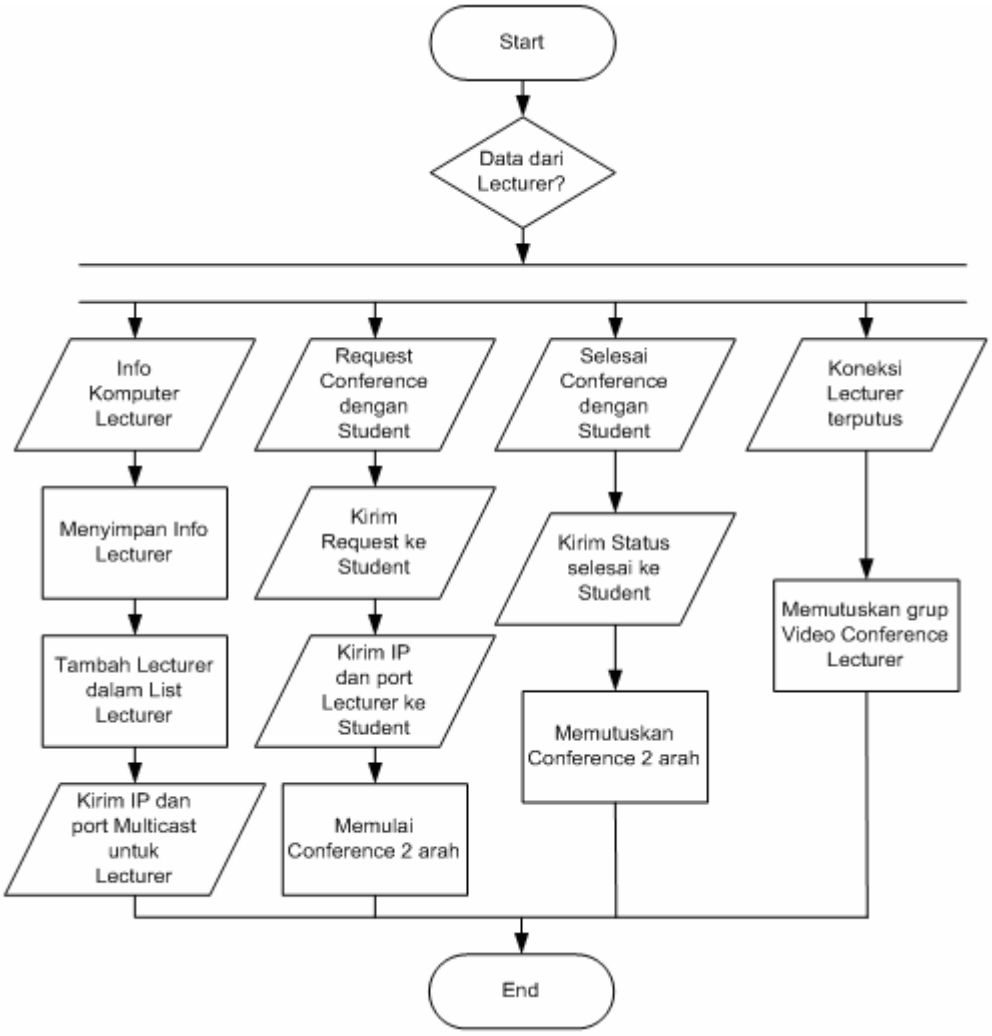
	melakukan koneksi ulang dan harus memilih pengajar lain yang masih terkoneksi atau keluar dari aplikasi. "SEROUT" & Chr$(2) & "server " & temp & " disonnected" & Chr$(4)
“VIDCHT”	Protokol ini digunakan pada saat aplikasi mahasiswa ingin melakukan video *conference* dengan pengajar. Aplikasi *server* akan mengirimkan protokol beserta *list* mahasiswa yang ingin melakukan video *conference* kepada pengajar. Aplikasi pengajar yang mendapat protokol ini maka *list user* dari mahasiswa yang bertanya akan ditandai dengan warna yang berbeda. Pengajar akan melakukan pemilihan salah satu mahasiswa untuk melakukan video *conference* "VIDCHT" & Chr$(2) & Client Nickname & vbcrlf & Client IP & Chr$(4)
"VIDENA"	Protokol ini digunakan apabila *list* mahasiswa yang ingin bertanya atau melakukan video *conference* dengan pengajar telah dipilih oleh pengajar. Aplikasi *server* akan mengirimkan protokol ini kepada mahasiswa yang dipilih dengan menambahkan alamat IP pengajar dan *port* yang digunakan. Protokol ini akan mengaktifkan koneksi video mahasiswa sehingga pengajar dan mahasiswa dapat melakukan video *conference*. "VIDENA" & Chr$(2) & Nickname Client & vbcrlf & IP Server

	& Chr$(4)
"VIDCLS"	Protokol ini digunakan pada saat aplikasi pengajar telah selesai mengadakan video *conference* dengan mahasiswa yang dipilih. Aplikasi *server* akan mengirimkan protokol ini kepada aplikasi mahasiswa yang dipilih. Protokol ini akan menghentikan pengiriman video dan suara dari aplikasi mahasiswa kepada aplikasi pengajar. "VIDCLS" & Chr$(2) & clientIp & vbcrlf & clientPort & Chr$(4)

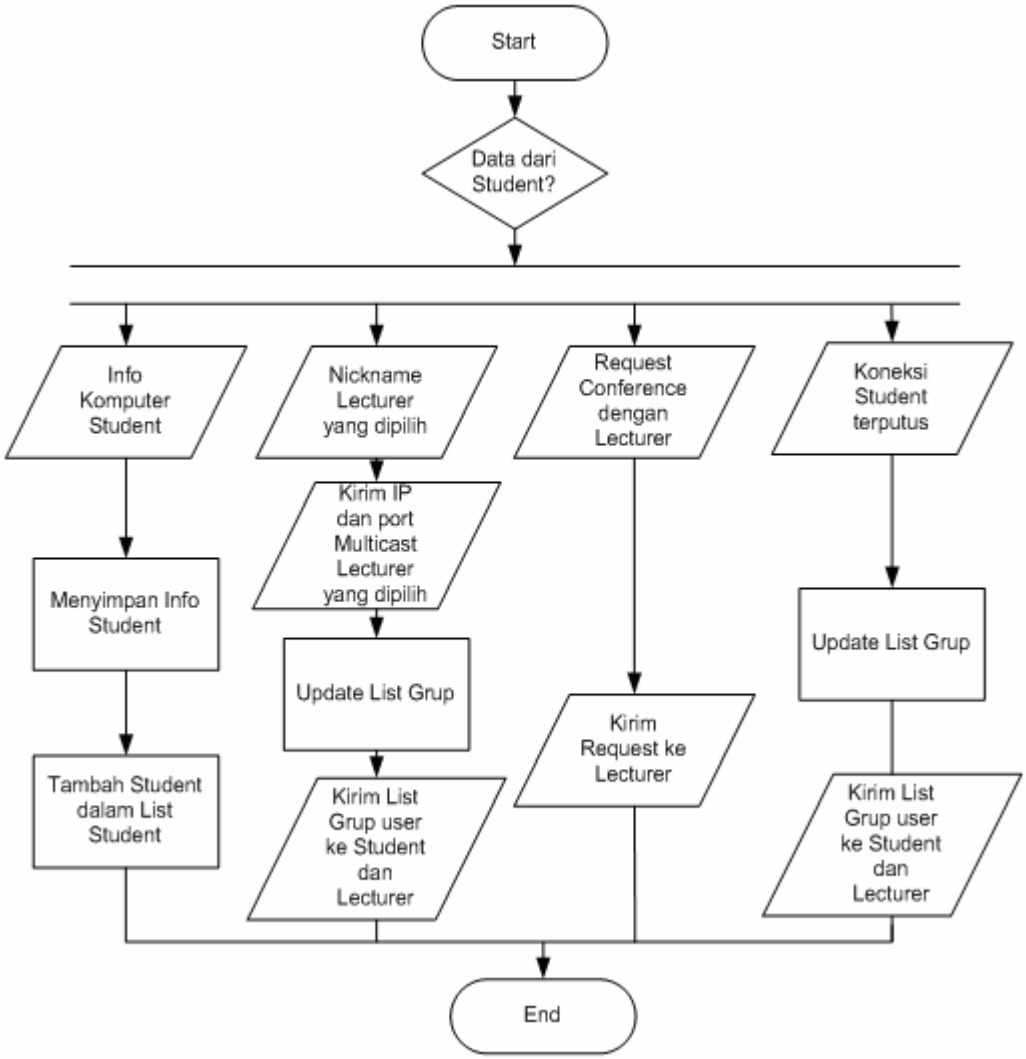
Pertama kali dijalankan maka aplikasi *server* harus diberikan *input* berupa *nickname* dan *port* yang akan digunakan secara manual, setelah itu *server* akan aktif dan menunggu koneksi. Jika terdapat koneksi maka *server* akan melakukan pengecekan dari protokol yang dikirimkan oleh *user*, dari protokol ini akan diketahui siapa yang terkoneksi. Jika pengajar maka akan dilakukan pengecekan data dari pengajar, sebaliknya jika mahasiswa akan dilakukan pengecekan data dari mahasiswa, dan apabila jika yang terkoneksi bukan keduanya maka data tidak akan diproses. Apabila salah satu *user* baik pengajar ataupun mahasiswa telah terkoneksi maka *server* akan menunggu koneksi selanjutnya, sehingga *server* dapat memproses lebih dari satu *user* yang terkoneksi. Berikut ini adalah diagram alir program *server*.



Gambar 3.9 Diagram alir utama *server*

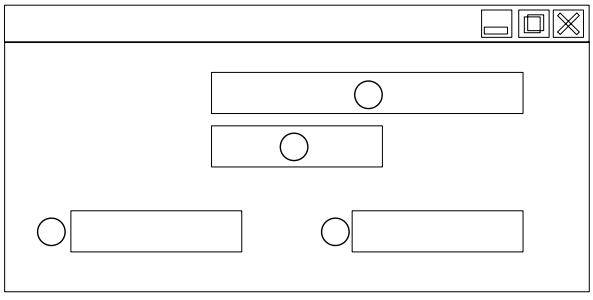


Gambar 3.10 Diagram alir *server* proses data pengajar



Gambar 3.11 Diagram alir *server* proses data mahasiswa

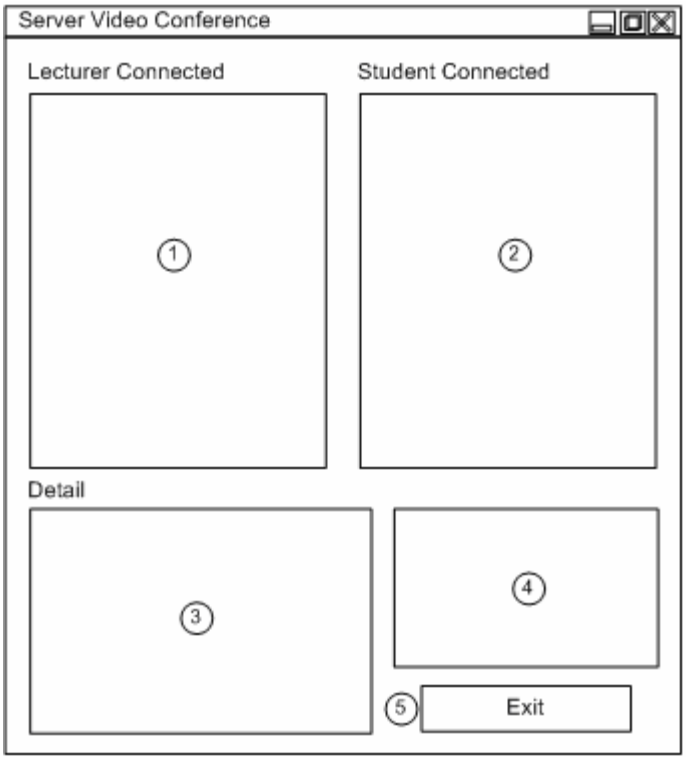
Berikut ini adalah rancangan GUI dari *server* :



Gambar 3.12 Rancang bangun *server* bagian 1

Keterangan gambar 3.12 :

1. *Input nickname server* yang akan digunakan
2. *Input* nomor *port* yang akan digunakan
3. Tombol *exit* untuk keluar dari aplikasi
4. Tombol *connect* berguna untuk mengaktifkan *server* apabila *user* telah selesai memasukkan data yang *user* inginkan



Gambar 3.13 Rancang bangun *server* bagian 2

Keterangan gambar 3.13 :

1. *List lecturer connect*ed akan ditampilkan *nickname* pengajar-pengajar yang aktif atau terkoneksi ke *server*
2. *List student connect*ed akan ditampilkan *nickname* mahasiswa-mahasiswa yang terkoneksi ke *server*

3. *User detail* akan menampilkan *detail user* baik pengajar maupun mahasiswa
4. *Multicast* grup *user* akan menampilkan siapa saja yang tergabung dalam grup *multicast*
5. Tombol *exit* digunakan untuk menghentikan grup *conference* dan keluar dari program *server*

3.4.2 Aplikasi Pengajar

Aplikasi pengajar saat diaktifkan akan terkoneksi langsung dengan *server* dengan menggunakan koneksi secara *unicast* dengan protokol TCP. Aplikasi pengajar langsung terkoneksi dengan *server* sehingga dapat dilakukan pengaturan secara otomatis dan tidak perlu konfigurasi yang terlalu banyak. Aplikasi pengajar akan mengirimkan data dengan *header* tertentu kepada *server* agar dapat berkomunikasi dengan mahasiswa, perintah yang digunakan dapat dilihat pada tabel 3.2 dibawah ini.

Tabel 3.2 Data perintah pada aplikasi pengajar (pengajar ke server atau mahasiswa)

Protokol	Keterangan
"SERCON"	Aplikasi pengajar saat pertama kali terkoneksi dengan *server* akan mengirimkan protokol ini kepada *server*, dimana menandakan bahwa pengajar terkoneksi. Protokol ini dikrimkan pengajar disertai dengan beberapa data yang akan disimpan oleh *server*. Data yang dikirimkan berupa status pengajar, *nickname* pengajar yang digunakan, IP komputer pengajar, dan nama komputer

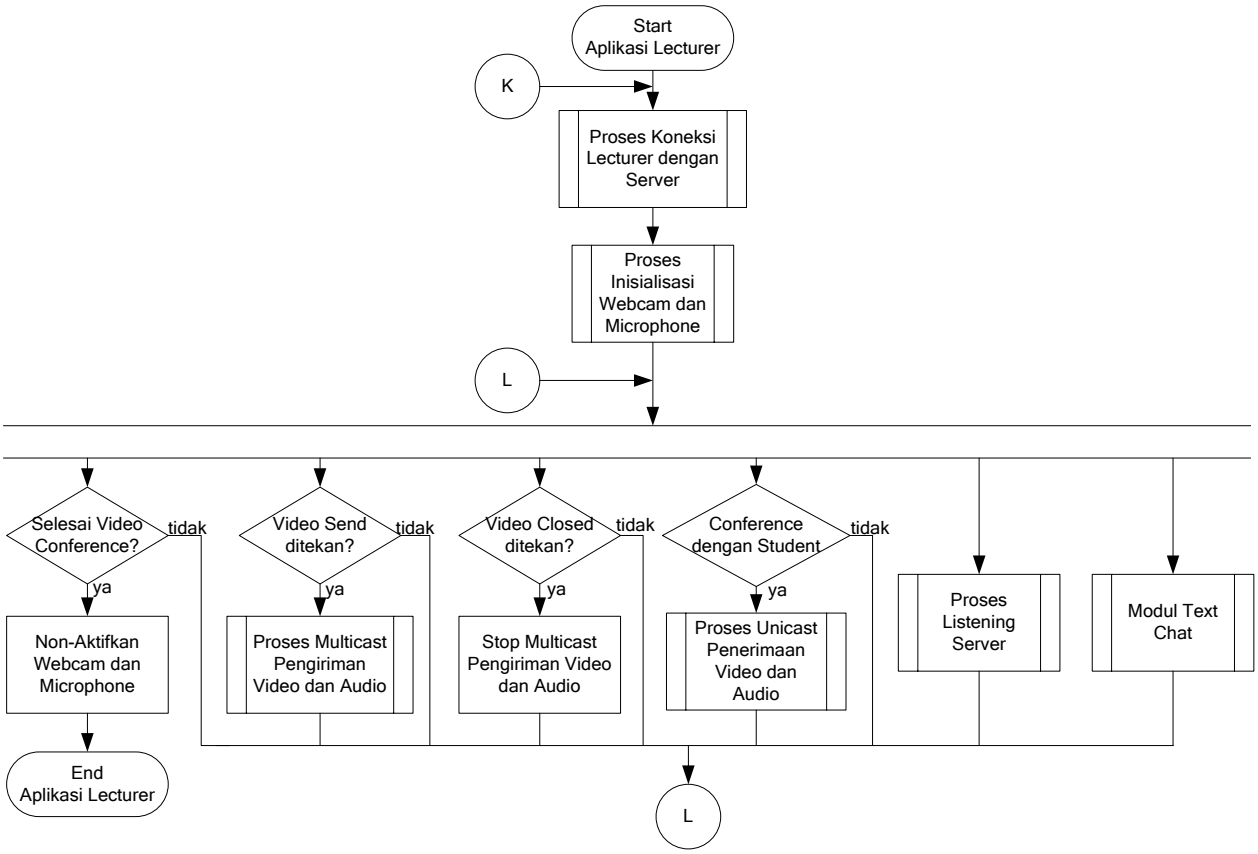
	pengajar. Aplikasi *server* akan melakukan pengecekan *nickname* dimana tidak boleh terdapat *nickname* yang sama, apabila sama maka pengajar akan diberikan perintah untuk mengganti *nickname* hingga berbeda. "SERCON" & Chr$(2) & Status & vbcrlf & NickName & vbcrlf & IP & vbcrlf & ComputerName & Chr$(4)
“VIDENA”	Protokol ini dikirimkan oleh aplikasi pengajar ke *server* dengan *nickname* mahasiswa yang dipilih oleh pengajar untuk melakukan video *conference*. *Server* akan memproses data ini dan mengirimkan protokol yang sama dengan alamat IP pengajar dan *port* yang digunakan. Aplikasi pengajar akan mengaktifkan form yang digunakan untuk menampilkan video dan suara dari mahasiswa. Aplikasi mahasiswa akan melakukan pengiriman video dan suara ke pengajar, hal ini akan membuat pengajar dan mahasiswa dapat melakukan video *conference*. "VIDENA" & Chr$(2) & IP Server & vbcrlf & Port & vbcrlf & IP Client & Chr$(4)
“VIDCLS”	Protokol ini digunakan pada saat pengajar telah selesai melakukan video *conference* dengan mahasiswa yang dipilih. Aplikasi pengajar akan mengirimkan protokol ini beserta *nickname* mahasiswa kepada *server*. Informasi ini akan diproses oleh aplikasi *server* yang akan meneruskan kepada mahasiswa untuk menghentikan pengiriman video dan suara kepada pengajar.

	Aplikasi mahasiswa yang mendapatkan protokol ini akan melakukan penghentian pengiriman data suara dan video. "VIDCLS" & Chr$(2) & clientIp & vbcrlf & clientPort & Chr$(4)
“MSG”	Protokol ini sedikit berbeda dengan beberapa protokol diatas, protokol ini dikirimkan oleh pengajar kepada mahasiswa tanpa melalui *server*. Protokol ini dikirimkan melalui alamat IP *multicast* dan *port* untuk teks *chat*. Protokol ini dikirimkan setiap kali pengajar melakukan pengiriman teks, dimana protokol akan dikirimkan dengan teks yang diketikkan oleh pengajar. "MSG" & Chr$(2) & ComputerIP & vbcrlf & Message & Chr$(4)

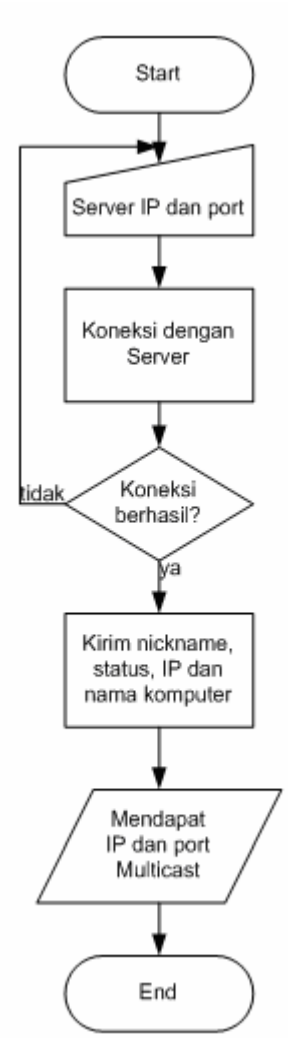
Aplikasi pengajar saat pertama kali diaktifkan harus diberikan *input nickname*, ip *server* dan *port* secara manual. Aplikasi kemudian akan melakukan koneksi dengan *server*, apabila ip *server* dan *port* yang di*input* benar maka aplikasi akan terkoneksi dengan *server*, jika tidak terkoneksi maka kembali harus dilakukan *input* ip *server* dan *port* hingga terkoneksi. Jika telah terkoneksi dengan *server* maka aplikasi pengajar akan mendapatkan alamat IP *multicast* dan *port* yang akan digunakan untuk mengirimkan video dan suara serta teks. Aplikasi pengajar akan melakukan inisialisasi *webcam* dan mikrofon agar dapat mengirimkan gambar dan suara ke mahasiswa. Proses akan dilanjutkan secara paralel, dimana akan terdapat beberapa proses diantaranya adalah proses mendengarkan data dari *server*, data dari *server* akan diproses sesuai perintah yang dikirimkan. Aplikasi akan mengirimkan suara dan video, jika dilakukan penekanan tombol pengiriman video. Terdapat proses pengiriman dan penerimaan teks *chat*, dimana saat melakukan

video *conference* aplikasi juga dapat melakukan teks *chat*. Apabila telah selesai melakukan *conference* ataupun terputus dari *server* maka akan dilakukan penonaktifan *webcam* dan mikrofon, dan aplikasi akan ditutup.

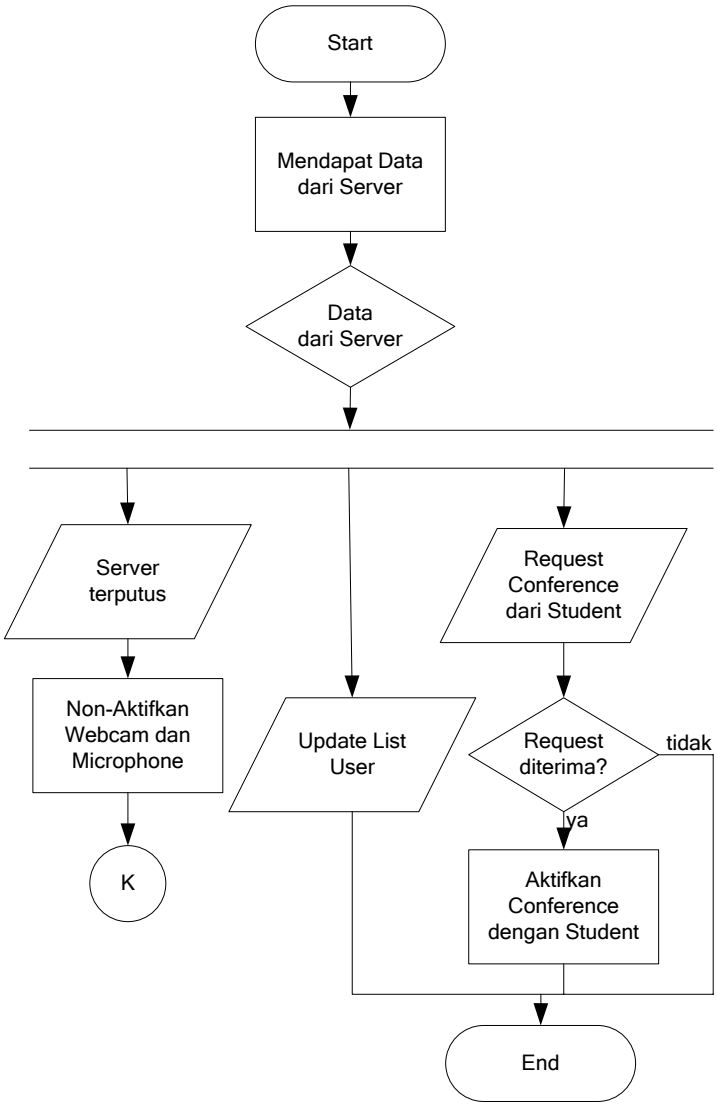
Berikut diagram alir untuk aplikasi pengajar:



Gambar 3.14 Diagram alir utama pengajar

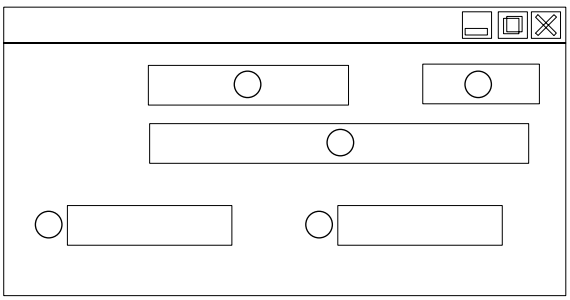


Gambar 3.15 Diagram alir koneksi pengajar dengan *server*



Gambar 3.16 Diagram alir pengajar *list*ening *server*

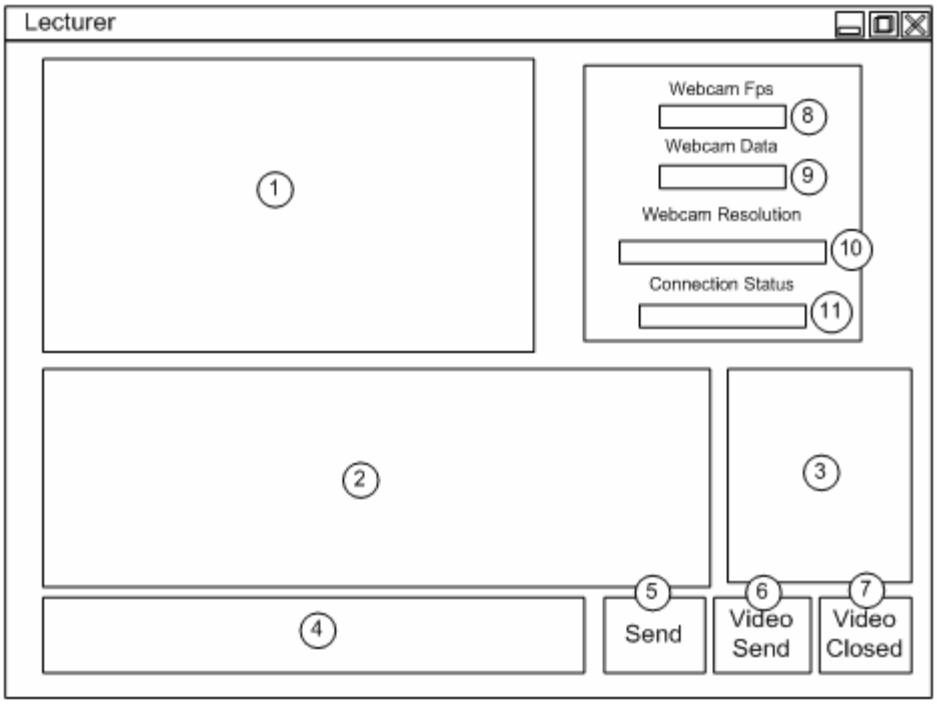
Berikut ini adalah rancangan GUI dari aplikasi pengajar:



Gambar 3.17 Rancang bangun pengajar bagian 1

Keterangan gambar 3.17 :

1. *Input* IP komputer *server*
2. *Input* nomor *port* komputer *server*
3. Input *nickname* pengajar yang akan digunakan dalam *conference*
4. Tombol *exit* berguna untuk keluar dari aplikasi
5. Tombol *connect* berguna untuk terkoneksi ke *server*

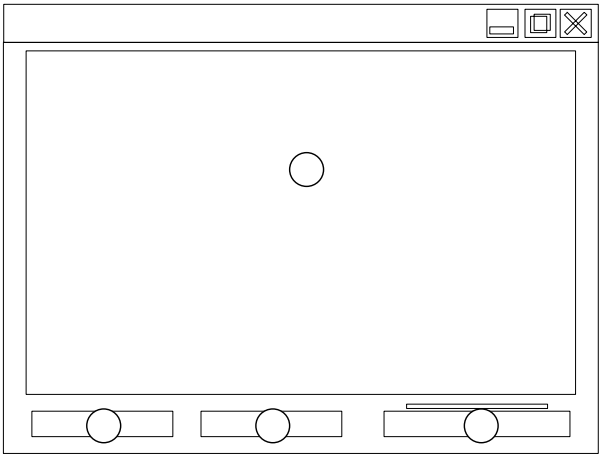


Gambar 3.18 Rancang bangun pengajar bagian 2

Keterangan gambar 3.18 :

1. Tampilan video dari *webcam* pengajar
2. Teks *chat* history berisi *log* dari teks *chat* yang terjadi dari awal aplikasi dijalankan serta keterangan yang dikirimkan *server*
3. *Multicast* grup *user* menampilkan siapa saja yang tergabung dalam grup *multicast*

4. *Input* teks yang akan dikirimkan
5. Tombol *send* untuk mengirimkan teks
6. Tombol video *send* untuk mengaktifkan pengiriman video dan suara secara *multicast*
7. Tombol video *closed* untuk menghentikan pengiriman video dan suara secara *multicast*
8. Besar *frame* per *second* video yang dikirimkan oleh aplikasi
9. Besar data dari *webcam* yang diterima
10. Ukuran resolusi *webcam* yang digunakan aplikasi
11. Status koneksi aplikasi. Status *connected* maka aplikasi mengirimkan video dan suara secara *multicast*, status *disconnected* maka aplikasi tidak melakukan pengiriman video dan suara.



Gambar 3.19 Rancang bangun pengajar bagian 3

Keterangan gambar 3.19 :

1. Video dari mahasiswa yang melakukan conference dengan pengajar
2. Besar *frame* per *second* dari video yang diterima

3. Besar data video yang diterima
4. Status koneksi menunjukkan koneksi pengajar dengan mahasiswa

3.4.3 Aplikasi Mahasiswa

Aplikasi mahasiswa saat diaktifkan akan terkoneksi langsung dengan *server* dengan menggunakan koneksi secara *unicast* dengan protokol TCP. Aplikasi mahasiswa akan mengirimkan data dengan *header* tertentu kepada *server* agar dapat berkomunikasi dengan pengajar, perintah yang digunakan dapat dilihat pada tabel 3.3 dibawah ini.

Tabel 3.3 Data perintah pada aplikasi mahasiwa (mahasiswa ke server atau pengajar)

Protokol	Keterangan
“CLICON”	Aplikasi mahasiswa saat pertama kali aktif akan mengirimkan protokol ini ke *server*, protokol ini menandakan bahwa terdapat mahasiswa yang terkoneksi. Protokol ini dikrimkan mahasiswa disertai dengan beberapa data yang akan disimpan oleh *server*. Data yang dikirimkan berupa status mahasiswa, *nickname* mahasiswa yang digunakan, IP komputer mahasiswa, dan nama komputer mahasiswa. Aplikasi *server* akan melakukan pengecekan *nickname* dimana tidak boleh terdapat *nickname* yang sama, apabila sama maka mahasiswa akan diberikan perintah untuk mengganti *nickname* hingga berbeda. "CLICON" & Chr$(2) & Status & vbcrlf & NickName & vbcrlf & IP & vbcrlf & ComputerName & Chr$(4)

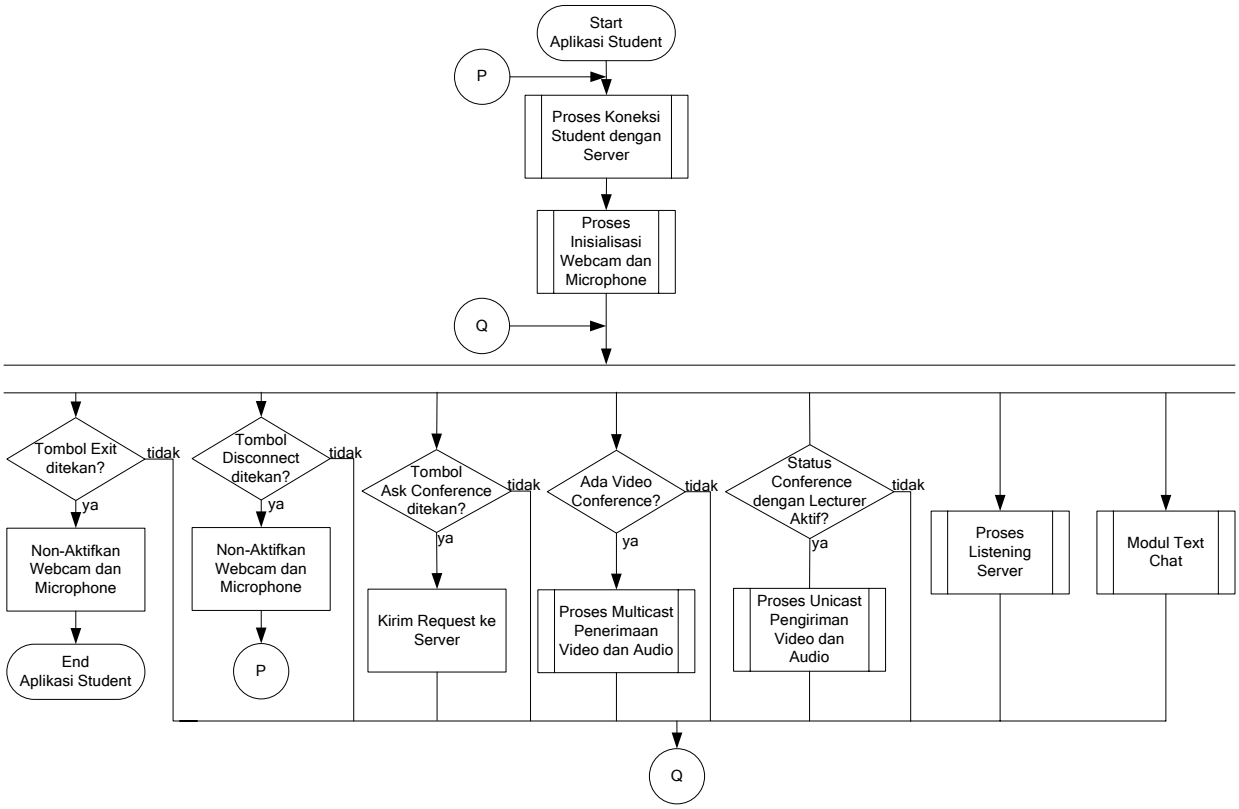
“CLICHS”	Aplikasi mahasiswa saat telah terkoneksi dan data telah disimpan oleh *server* maka aplikasi *server* akan mengirimkan *list* pengajar ke mahasiswa untuk dipilih oleh mahasiswa. Protokol ini digunakan pada saat mahasiswa ingin memilih pengajar untuk melakukan video *conference*. Aplikasi mahasiswa akan mengirimkan protokol ini beserta *nickname* pengajar yang dipilih. Informasi ini akan diproses oleh aplikasi *server* yang kemudian akan memberikan alamat IP *multicast* serta *port* kepada mahasiswa, sehingga mahasiswa dan pengajar akan tergabung dalam grup *multicast* yang sama dan dapat melakukan video *conference*. "CLICHS" & Chr$(2) & Status & vbcrlf & Server Choose & Chr$(4)
“ASKVID”	Pada saat terjadinya video *conference* apabila mahasiswa ingin melakukan video *conference* dengan pengajar maka akan mengirimkan protokol ini. Protokol ini akan dikirimkan ke *server*, dan akan diproses oleh *server* yang akan memberikan tanda ke pengajar bahwa terdapat satu atau beberapa mahasiswa yang ingin melakukan video *conference*. Pengajar akan melihat *nickname* mahasiswa yang bertanya berubah warna di *list user*. Pengajar jika ingin melakukan video *conference* dengan mahasiswa maka akan mengirimkan perintah ke *server*, yang kemudian akan mengaktifkan video di kedua sisi baik pengajar dan mahasiswa sehingga memungkinkan terjadinya video *conference*.

	"ASKVID" & Chr$(2) & Chr$(4)
“MSG”	Sama halnya dengan aplikasi pengajar, protokol ini sedikit berbeda dengan beberapa protokol diatas. Protokol ini dikirimkan oleh mahasiswa kepada pengajar tanpa melalui *server*. Protokol ini dikirimkan melalui alamat IP *multicast* dan *port* untuk teks *chat*. Protokol ini dikirimkan setiap kali mahasiswa melakukan pengiriman teks, dimana protokol akan dikirimkan dengan teks yang diketikkan oleh mahasiswa. "MSG" & Chr$(2) & ComputerIP & vbcrlf & Message & Chr$(4)

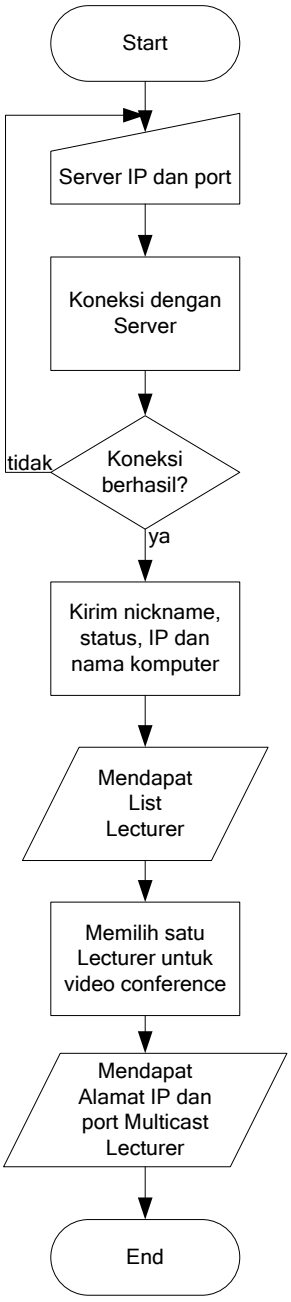
Aplikasi mahasiswa saat pertama kali diaktifkan harus diberikan *input nickname*, ip *server* dan *port* secara manual. Aplikasi kemudian akan melakukan koneksi dengan *server*, apabila ip *server* dan *port* yang di*input* benar maka aplikasi akan terkoneksi dengan *server*, jika tidak terkoneksi maka kembali harus dilakukan *input* ip *server* dan *port* hingga terkoneksi. Jika telah terkoneksi dengan *server* maka aplikasi mahasiswa akan mendapatkan *list* pengajar yang saat itu sedang terkoneksi ke *server*. Mahasiswa akan memilih salah satu diantara *list* pengajar, setelah melakukan pemilihan maka aplikasi akan mengirimkan pengajar yang dipilih ke *server* dan mahasiswa akan mendapatkan alamat IP *multicast* dan *port* yang digunakan oleh pengajar, sehingga mahasiswa dan pengajar akan tergabung dalam satu grup *multicast* yang sama. Aplikasi akan melakukan inisialisasi *webcam* dan mikrofon agar dapat mengirimkan gambar dan suara ke pengajar. Proses akan dilanjutkan secara paralel, dimana akan terdapat beberapa proses diantaranya adalah proses mendengarkan data dari *server*, data dari *server* akan

diproses sesuai dengan perintah yang dikirimkan. Aplikasi akan menerima suara dan video dari *server*. Proses yang lain adalah pengiriman dan penerimaan teks *chat*, seperti halnya aplikasi pengajar. Apabila telah selesai melakukan *conference* ataupun aplikasi terputus dari *server* maka akan dilakukan penonaktifan *webcam* dan mikrofon, dan aplikasi akan ditutup. Aplikasi akan melakukan proses dari awal jika koneksi aplikasi mahasiswa dengan pengajar terputus, dimana *user* harus melakukan pemilihan pengajar lain yang saat itu terkoneksi.

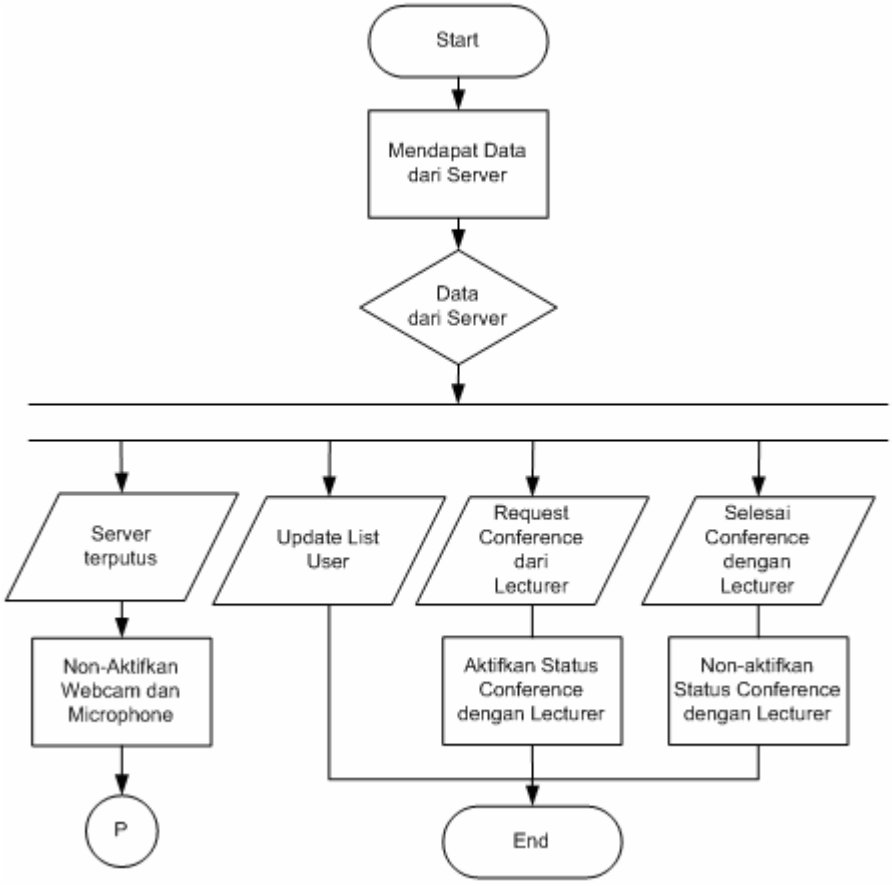
Berikut diagram alir untuk aplikasi mahasiswa:



Gambar 3.20 Diagram alir utama mahasiswa

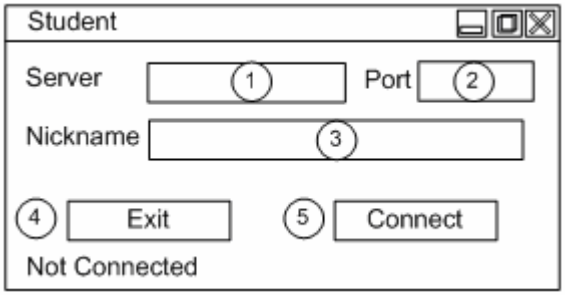


Gambar 3.21 Diagram alir koneksi mahasiswa dengan *server*



Gambar 3.22 Diagram alir mahasiswa *list*ening *server*

Berikut ini adalah rancangan GUI dari aplikasi mahasiswa:

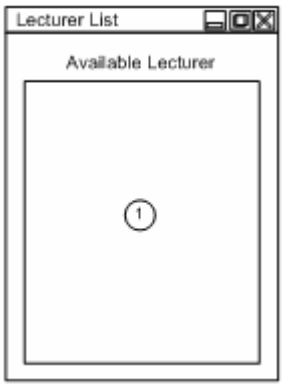


Gambar 3.23 Rancang bangun mahasiswa bagian 1

Keterangan gambar 3.23 :

1. *Input* IP komputer *server*
2. *Input* nomor *port* komputer *server*

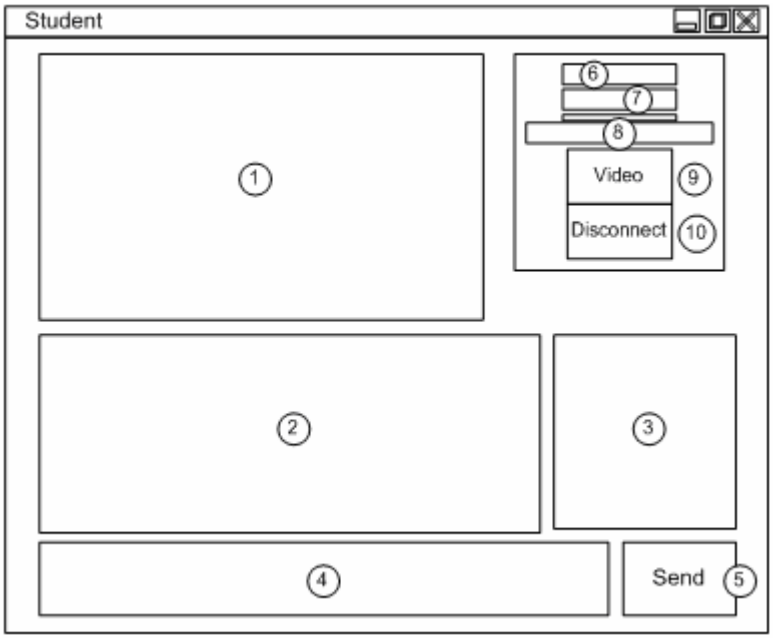
3. Input *nickname* mahasiswa yang akan digunakan dalam *conference*
4. Tombol *exit* berguna untuk keluar dari aplikasi
5. Tombol *connect* berguna untuk terkoneksi ke *server*



Gambar 3.24 Rancang bangun mahasiswa bagian 2

Keterangan gambar 3.24 :

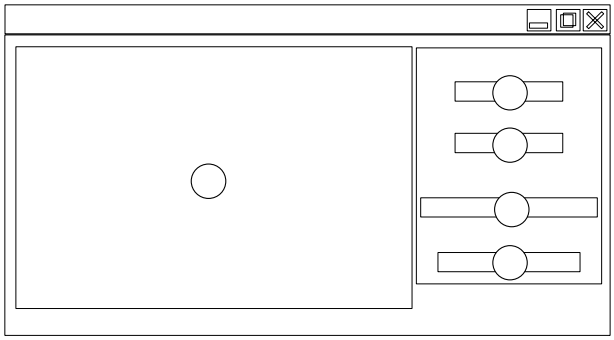
1. *List connect*ed *lecturer* merupakan daftar *nickname* pengajar-pengajar yang terkoneksi ke *server*



Gambar 3.25 Rancang bangun mahasiswa bagian 3

Keterangan gambar 3.25 :

1. Tampilan video pengajar yang diterima oleh mahasiswa
2. Teks *chat history* berisi *log* dari teks *chat* yang terjadi dari awal aplikasi dijalankan serta keterangan yang dikirimkan *server*
3. *Multicast* grup *user* menampilkan siapa saja yang tergabung dalam grup *multicast*
4. *Input* teks yang akan dikirimkan
5. Tombol *send* untuk mengirimkan teks
6. Besar *frame* per *second* video yang dikirimkan oleh aplikasi
7. Besar data dari *webcam* yang diterima setiap waktu
8. Status koneksi aplikasi. Status *connected* maka aplikasi mengirimkan video dan suara secara *multicast*, status *disconnected* maka aplikasi tidak melakukan pengiriman video dan suara.
9. Tombol video digunakan untuk melakukan video *conference* dengan pengajar
10. Tombol dis*connect* untuk menghentikan penerimaan video dari pengajar dan kembali pada tampilan pemilihan pengajar



Gambar 3.26 Rancang bangun mahasiswa bagian 4

Keterangan gambar 3.26 :

1. Tampilan *webcam* yang dimiliki oleh mahasiswa
2. Besar *frame* per *second* video yang dikirimkan
3. Besar data dari *webcam* yang diterima
4. Ukuran resolusi *webcam* yang digunakan
5. Status koneksi mahasiswa dengan pengajar