BAB III
ANALISIS DAN PERANCANGAN SISTEM

Analisis dan perancangan sistem memerlukan tahapan yang sistematis untuk mendapatkan rancangan sistem yang baik dan sesuai dengan kegunaan dan tujuannya. Tahap awal dari analisis adalah menganalisa kebutuhan-kebutuhan sistem mulai dari kebutuhan pengguna, kebutuhan non fungsional, dan kebutuhan fungsional. Sedangkan untuk tahap perancangan aplikasi yaitu perancangan *database* dan perancangan antarmuka.

3.1 Analisis Sistem

Analisis merupakan suatu tahapan pemahaman terhadap sistem atau aplikasi yang sedang berjalan maupun yang akan dibuat. Tahapan analisis bertujuan untuk mengetahui mekanisme atau prosedur kerja dari proses yang sedang berjalan maupun yang akan dibuat. Tahap analisis sistem ini merupakan tahap yang sangat kritis dan sangat penting, karena kesalahan di dalam tahap ini akan menyebabkan juga kesalahan di tahap selanjutnya. Tujuan dari tahapan-tahapan tersebut adalah untuk identifikasi elemen dan bobot penilaian kinerja pegawai, batasan sistem dan hubungan antar komponen sistem. Hasil akhir dari analisis sistem adalah deskripsi dan pemetaan sistem yang akan dibangun yang digambarkan dengan diagram alur.

3.2 Analisis Masalah

Dalam penelitian ini sebelum menentukan masalah pemantauan lalu lintas di bagian pelayanan lalu lintas PT.Jasa Marga(Persero), terlebih dahulu peneliti meng-observasi arsitektur sistem yang sedang berjalan. Di dalam arsitektur sistem pemantauan lalu lintas kendaraan terdapat unsur-unsur pendukung sistem sebagai berikut:

1. Infrastruktur jalur

   Infrastruktur jalur adalah perangkat pendukung pemantauan lalu lintas kendaraan yang ada di jalur tol. Seperti CCTV, konverter FO, kabel fiber optik.

2. Infrastruktur plaza

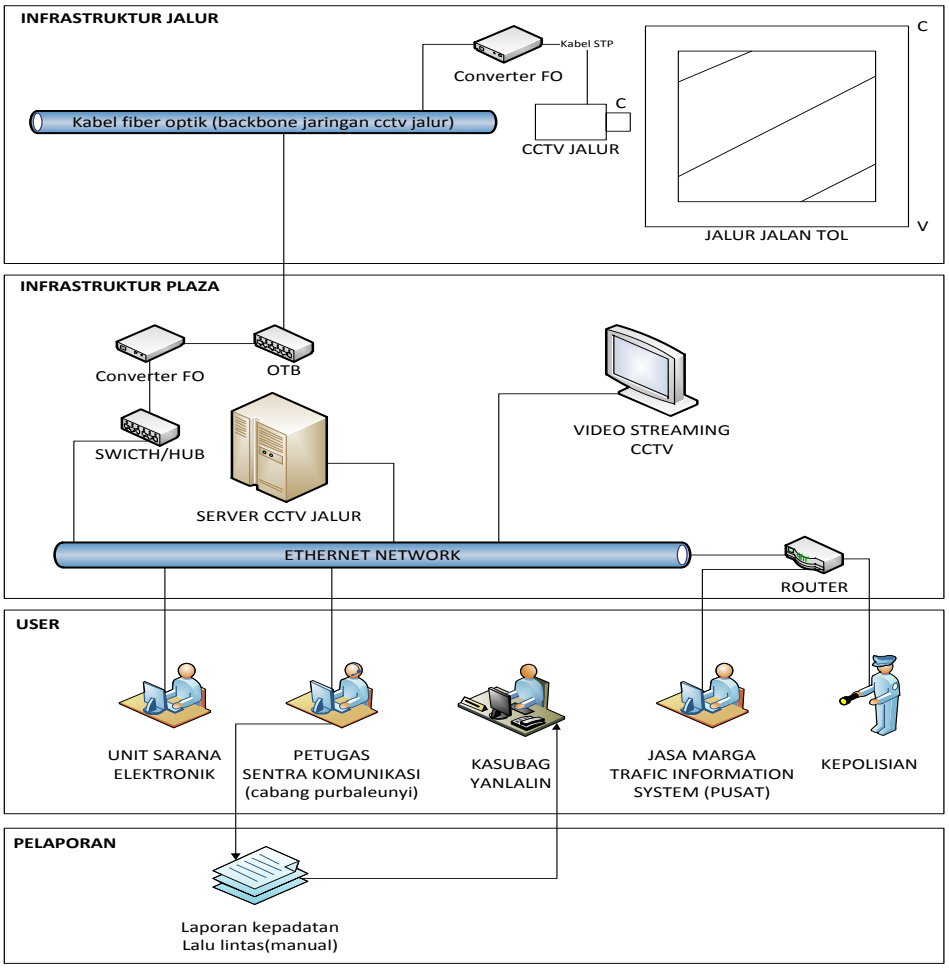
   Infrastruktur adalah perangkat pendukung pemantauan lalu lintas kendaraan yang ada kantor PT.Jasa Marga(Persero) cabang purbaleunyi. Seperti OTB, konverter FO, Swicth/HUB, server, monitor, router, jaringan LAN.

3. User

   User adalah pengguna dari sistem pemantauan lalu lintas. Seperti: petugas Sentra Komunikasi(SENKOM), Unit Sarlek, Kasubag Yanlalin, Jasa Marga Traffic Information System(JMTIS), Kepolisian.

4. Pelaporan

   Pelaporan kepadatan lalu lintas kendaraan hasil pemantauan yang dibuat oleh petugas Sentra Komunikasi(SENKOM) yang kemudian diserahkan kepada Kasubag Yanlalin.

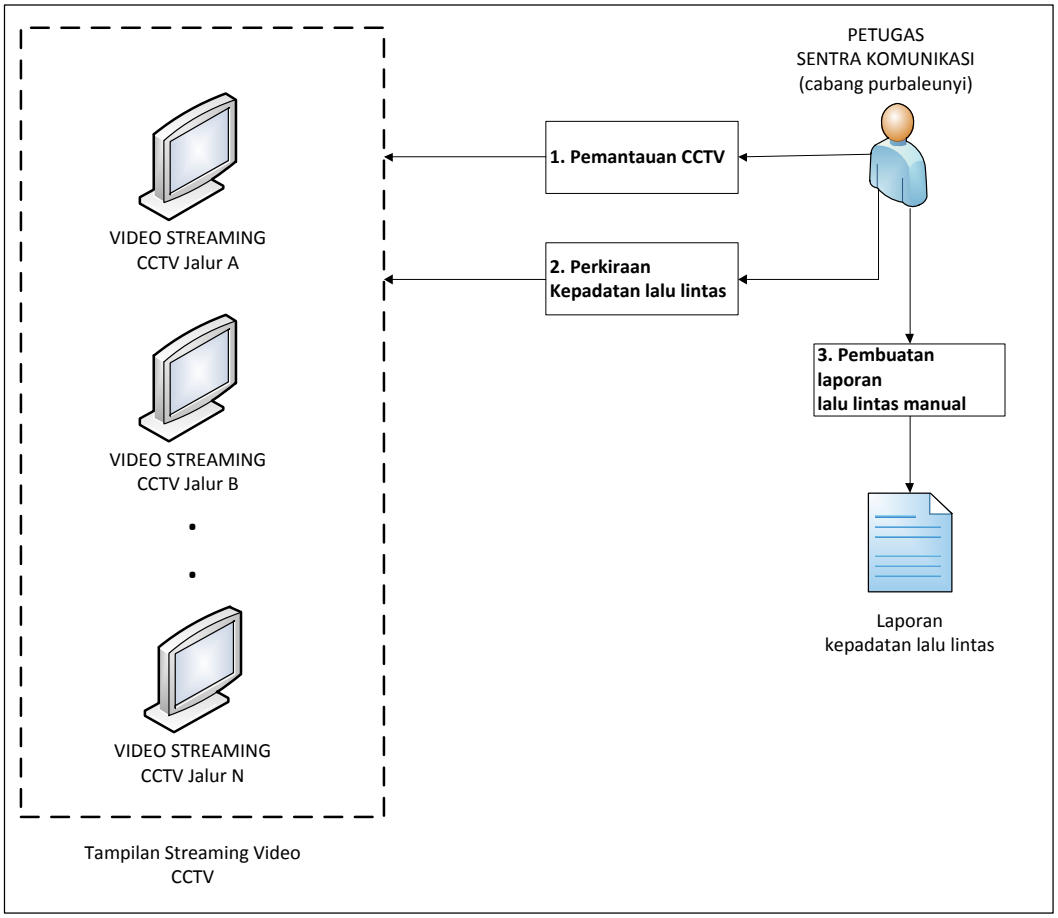


Gambar III.1 Arsitektur Sistem Yang Sedang Berjalan

Berdasarkan data yang diperoleh dari hasil penelitian, interview dan observasi langsung terhadap pihak bagian Pelayanan Lalu Lintas PT.Jasa Marga (Persero), didapati beberapa masalah dalam melakukan pemantauan kepadatan lalu lintas, yaitu:

1. Pemantauan kepadatan lalu lintas kendaraan oleh petugas Sentra Komunikasi(SENKOM) dilakukan secara visual melalui media *video streaming* CCTV jalur yang dipantau secara terus menerus. Dan untuk menentukan kepadatan lalu lintas kendaraan dengan perkiraan, sebagai acuan dari perkiraan yaitu melihat kendaraan yang telah memenuhi semua jalur jalan dan ujung dari kendaraan yang telah memanjang.
2. Apabila ada penambahan titik akses pemantauan lalu lintas di jalur jalan tol akan dibutuhkan penambahan petugas yang bertugas untuk pemantauan lalu lintas kendaraan dan penambahan ruangan yang semakin banyak untuk penempatan perangkat pemantauan lalu lintas.
3. Akses untuk mendapatkan informasi kepadatan lalu lintas kendaraan relatif lama dikarenakan petugas harus melihat secara visual melalui media streaming CCTV jalur dan memperkirakan kepadatan lalu lintas kendaraan.

Berikut adalah deskripsi sistem yang sedang berjalan:



Gambar III.2 Deskripsi Sistem Yang Sedang Berjalan

Untuk menyelesaikan permasalahan ini, penulis bermaksud untuk mengimplementasikan suatu sistem yang memberikan informasi tentang kepadatan lalu lintas secara otomatis. Sistem yang dibuat akan mengotomatisasi dalam pemantauan kepadatan lalu lintas. Hal ini akan mengurangi penambahan petugas dan penambahan ruang. Data volume kendaraan diperoleh melalui pengolahan data citra yang di ambil dari streaming video dari kamera pemantau yang berada di jalur jalan tol. Aplikasi yang dibangun berbasis dekstop dengan arsitektur jaringan komputer berbasis client-server.

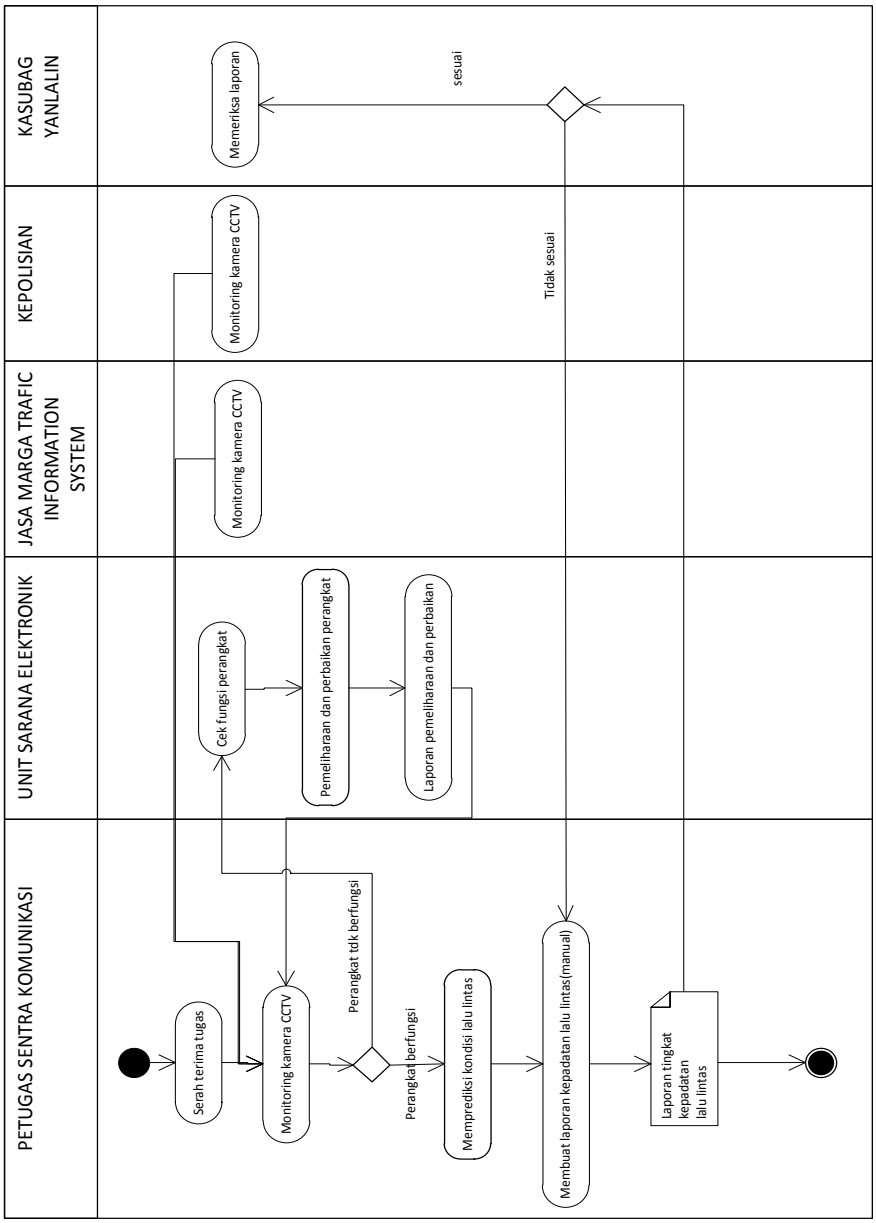
3.3 Analisis Prosedur

Analisis Prosedur atau analisis proses adalah tahapan yang memberi gambaran tentang sistem yang sedang berjalan. Analisis ini bertujuan untuk memberi gambaran yang lebih detail bagaimana cara kerja dari sistem yang

sedang berjalan. Analisa prosedur mekanisme pemantauan lalu lintas adalah sebagai berikut:

1. Petugas Sentra Komunikasi melakukan serah terima tugas dari petugas shift sebelumnya ke petugas shift selanjutnya.
2. Petugas Sentra Komunikasi melakukan pemantauan lalu lintas melalui media video streaming CCTV jalur yang dipantau terus menerus.
3. Petugas Sentra Komunikasi memperkirakan kondisi lalu lintas berdasarkan pemantauan secara visual dari media video streaming CCTV lajur.
4. Petugas Sentra Komunikasi Membuat laporan kepadatan lalu lintas yang ditulis manual pada form yang telah disediakan, dan menyerahkan laporan tersebut kepada Kasubag Yanlalin.
5. Kasubag Yanlalin memeriksa laporan kepadatan lalu lintas yang dibuat oleh Petugas Sentra Komunikasi. Apabila laporan tidak sesuai maka laporan tersebut diserahkan kembali untuk diperbaiki oleh petugas yang bersangkutan.
6. *Jasa Marga Traffic Information System*(JMTIS) melakukan pemantauan kepadatan lalu lintas sewaktu-waktu untuk mengecek kondisi lalu lintas.
7. Kepolisian adalah pihak lain yang telah bekerja sama dengan PT.Jasa Marga(Persero) yang melakukan pemantauan kepadatan lalu lintas sewaktu-waktu untuk mengecek kondisi lalu lintas.
8. Apabila terjadi kerusakan atau tidak berfungsinya perangkat pemantauan petugas sentra komunikasi melaporkan kepada unit sarana elektronik untuk diperbaiki.
9. Unit sarana elektronik adalah divisi yang bertugas untuk memelihara dan memperbaiki perangkat pemantauan lalu lintas.

Berikut adalah gambar prosedur mekanisme pemantauan lalu lintas:



Gambar III.3 Prosedur Mekanisme Pemantauan Lalu Lintas

3.3.1 Analisis Proses

Adapun proses yang ada pada sistem yang sedang berjalan adalah sebagai berikut:

1. Proses pemantauan lalu lintas melalui media video streaming CCTV jalur.
2. Proses perkiraan kepadatan lalu lintas kendaraan.
3. Proses pembuatan laporan kepadatan lalu lintas yang dilakukan secara manual dengan ditulis tangan pada form yang telah disediakan.

3.3.2 Analisis Sistem Yang Sedang Berjalan

Saat ini sistem pemantauan kepadatan lalu lintas masih bersifat manual sesuai dengan yang dijelaskan pada Analisis Prosedur. Sesuai dengan prosedur tersebut di atas, maka dibutuhkan sebuah aplikasi Sistem Pemantau Kepadatan Lalu Lintas Otomatis, untuk mempermudah dalam pemantauan kepadatan lalu lintas.

3.4 Analisis Sistem yang Akan Dibangun

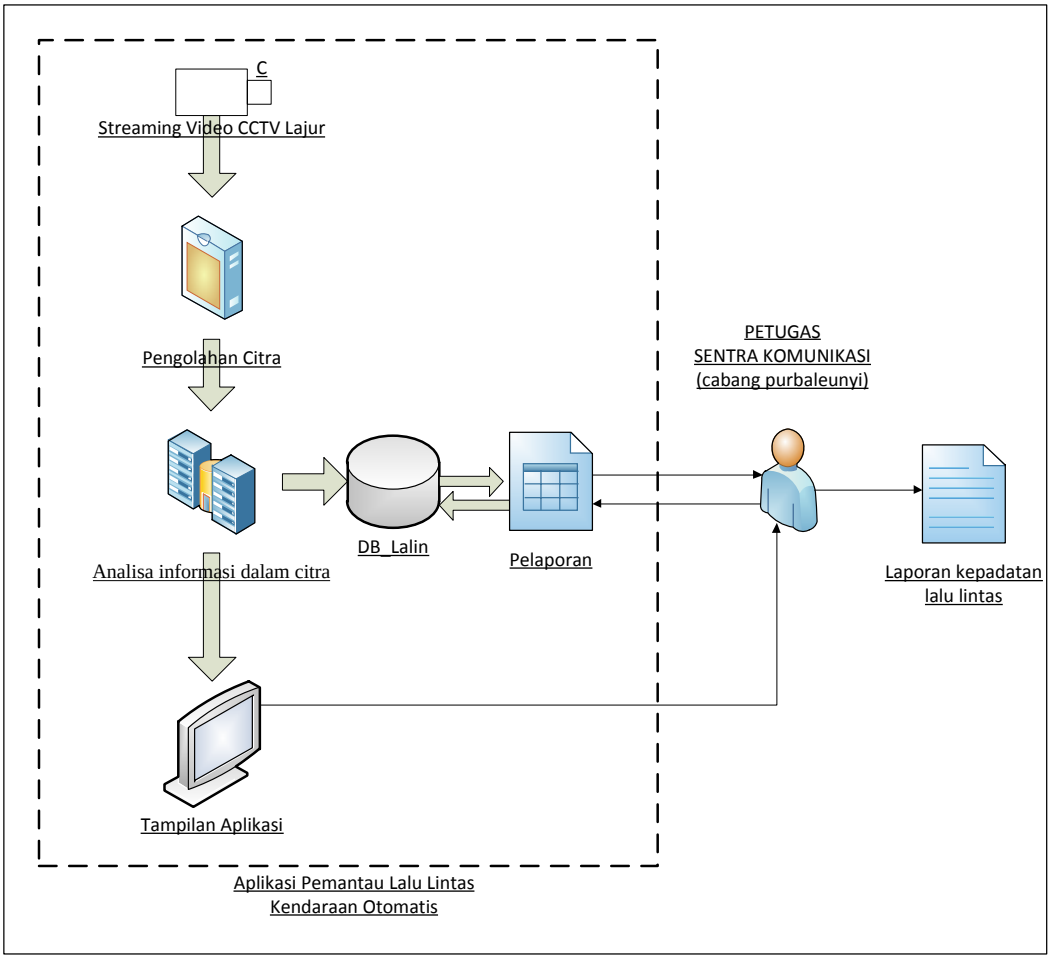
Sesuai dengan analisis sistem yang sedang berjalan saat ini, maka sistem baru yang akan dibangun merubah pemantauan manual yang dilakukan oleh petugas sentra komunikasi menjadi otomatis

3.4.1 Arsitektur Sistem yang Akan dibangun

Sistem yang dibangun terdiri dari subsistem-subsistem berikut:

1. Subsistem s*treaming* video CCTV
   Sumber video berasal dari kamera yang berada di berbagai titik di jalan tol. Subsistem ini sudah ada dan sekarang sedang berjalan.
2. Subsistem pengolahan citra
   Bagian pengolahan citra terdiri dari proses penentuan ROI, *gray-scaling*, NSSD, dan thresholding.
3. Subsistem analisa informasi dalam citra
   Pada bagian ini terdapat proses penghitungan jumlah kendaraan yang terpantau melalui kamera, penentunan tingkat kepadatan kendaraan, serta penyimpanan informasi ini ke dalam *database*.
4. Subsistem pelaporan
   Pada bagian pelaporan terdapat pembangkitan laporan yang dapat dipilih berdasarkan parameter waktu yang diperlukan.

Berikut adalah deskripsi dari aplikasi sistem pemantau kepadatan lalu lintas kendaraan yang akan dibangun:



Gambar III.4 Deskripsi Sistem Yang Akan Dibangun

Dalam pemantauan kepadatan lalu lintas kendaraan, petugas sentra komunikasi dibantu oleh aplikasi pemantauan lalu lintas sehingga petugas tidak terus-menerus melakukan pemantauan. Aplikasi pemantauan lalu lintas juga menyediakan fitur dalam pembuatan laporan kepadatan lalu lintas yaitu laporan kepadatan lalu lintas per jam dan laporan kepadatan lalu lintas per hari.

3.4.2 Analisis Proses Penghitungan Kendaraan

Pada bagian sebelumnya telah diterangkan bagian-bagian sistem yang akan dibangun. Secara detail, proses-proses dalam sistem pemantau kepadatan lalu lintas ini adalah sebagai berikut:

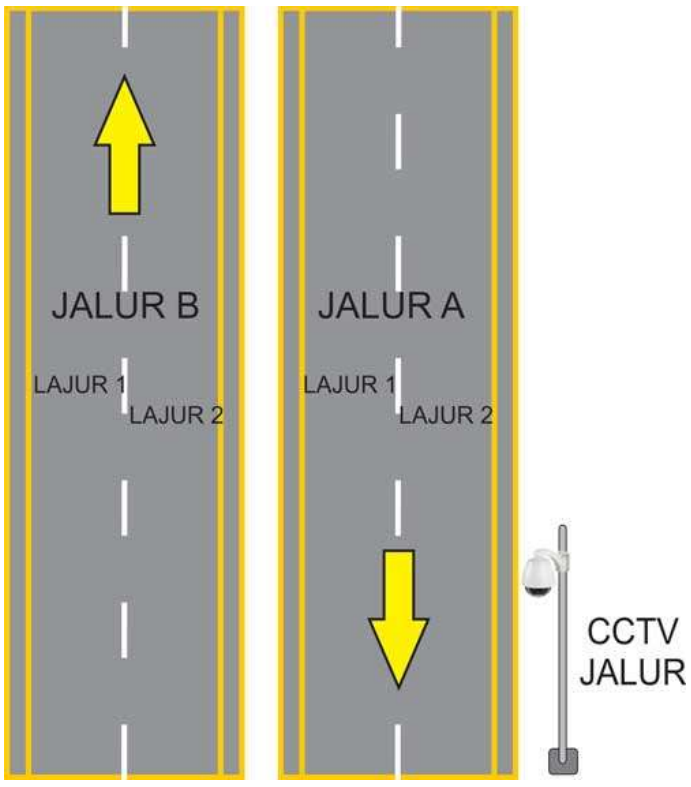
1. Proses Capture Kamera
1. Penentuan Region of Interest
2. Proses Grayscaling

3. Proses NSSD
4. Thresholding
5. Penghitungan kendaraan

Penjelasan lengkap untuk setiap tahapan akan dijelaskan pada bagian berikut.

3.4.2.1 Proses *Capture* Kamera

Proses *Capture* ini merupakan proses yang sangat penting, karena melalui proses inilah data input akan didapatkan untuk dilakukan proses penghitungan. Pada system realtime proses *Capture* sendiri akan terus dilakukan walaupun jalan dalam keadaan kosong. Kecepatan kamera untuk proses peng-*Capture*-an yang digunakan adalah 15fps. Adapun posisi untuk penempatan kamera yang digunakan dalam penelitian ini adalah berada di tepi sisi ruas jalan dengan ketinggian yang telah di tentukan dan kemiringan sekitar 75 derajat, untuk lebih jelasnya dapat dilihat dari gambar III.5:



Gambar III.5 Penempatan Posisi Kamera CCTV

Dalam penelitian ini hanya satu jalur jalan yang diambil yaitu jalur A, dan hasil *Capture* adalah sebagai berikut:



Gambar III.6 Hasil *Capture* Kamera

```
Procedure VideoCapture(mjpegSource : MJPEGStream)

Kamus
   height, width : integer
   URL : array[] of character
   lastFrame : Bitmap

struct MJPEGStream
   procedure VideoSource(URL : array[] of character)
   procedure Start()
   procedure Stop()
   procedure Bitmap()

Algoritma
   URL ← "http://202.90.195.131:218/axis-cgi/mjpg/video.cgi"
   mjpegSource.VideoSource(URL)
   mjpegSource.Start()
   if (mjpegSource <> NULL) then
      lastFrame ← mjpegSource.Bitmap
      height ← lastFrame.Height
      width ← lastFrame.Width
   EndIf
   mjpegSource.Stop()
```

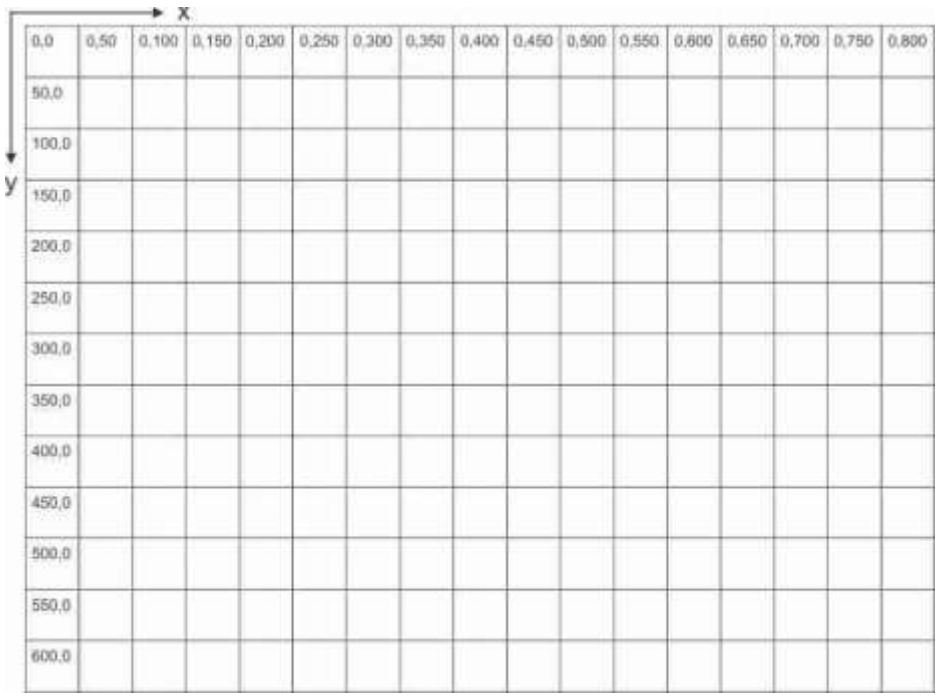
Gambar III.7 Algoritma Capture Kamera MJPEG(Stream)

3.4.2.2 Penentuan *Region of Interest*

Setelah proses *Capture* kamera proses selanjutnya adalah proses setting *Region of Interest* (ROI). ROI adalah daerah berbentuk persegi panjang dimana objek sering dijumpai di dalam citra. Objek ini dapat berupa kendaraan yang akan dihitung ataupun objek lain. Tujuan pembentukan daerah ini adalah untuk mempermudah pendeteksian objek di dalam citra. ROI ditentukan melalui dua tahapan proses. Adapun tahapan dari proses pembuatan ROI adalah sebagai berikut :

A. Tentukan dimensi citra

Proses penentuan ROI dalam tugas akhir ini pada mulanya dilakukan dengan menentukan dimensi citra. Dimensi citra adalah ukuran panjang dan lebar citra dalam ruang dua dimensi x dan y. Proses ini penting mengingat bisa saja dimensi citra yang akan diproses berbeda-beda. Dimensi citra dapat dengan mudah diperoleh dari data image yang akan diproses, yaitu properti panjang (image.height) dan lebar (image.width).



Gambar III.8 Jumlah piksel dalam *Region of Interest*

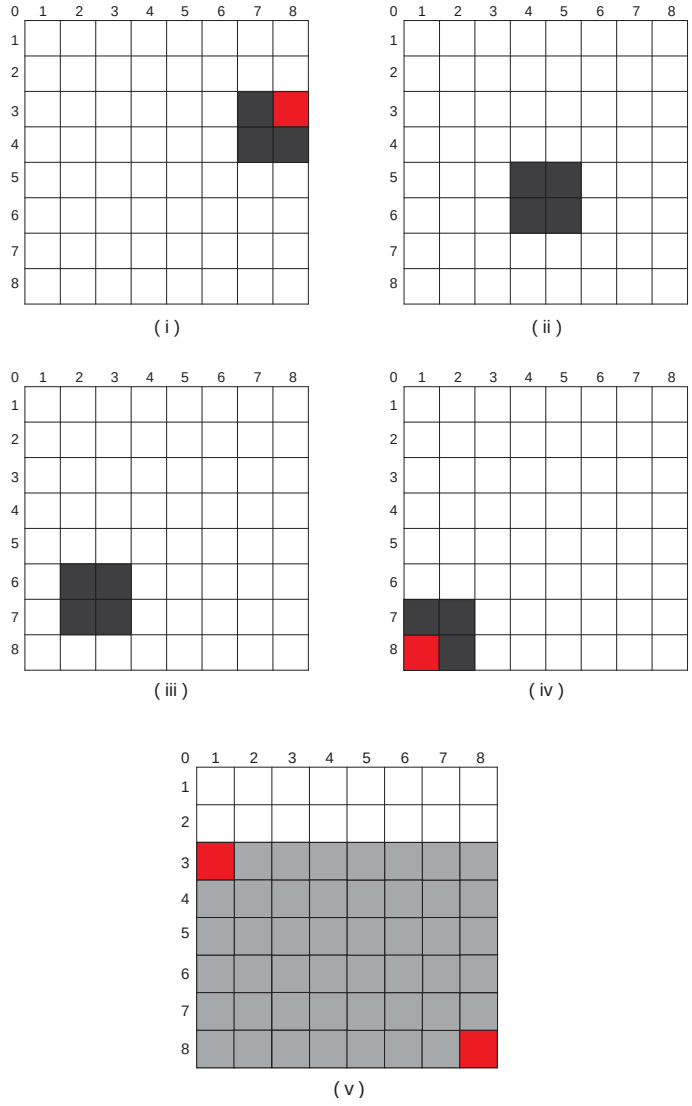
B. Tentukan titik-titik daerah objek

Dalam sebuah citra, bisa saja terdapat banyak objek di dalamnya. Untuk memudahkan identifikasi objek yang akan diproses maka dilakukan

penentuan daerah (*region*) yang paling potensial. Titik-titik ini ditentukan dengan mengamati pergerakan objek dalam citra. Prosesnya dapat dijelaskan sebagai berikut:

1. Dapatkan arah pergerakan objek dalam runtunan frame citra yang ada,
2. Dengan menggunakan referensi dimensi citra yang didapat pada bagian sebelumnya, tentukan empat titik untuk membangun daerah persegi panjang di dalam citra.

Misalnya terdapat runtunan citra seperti pada Gambar III.9:

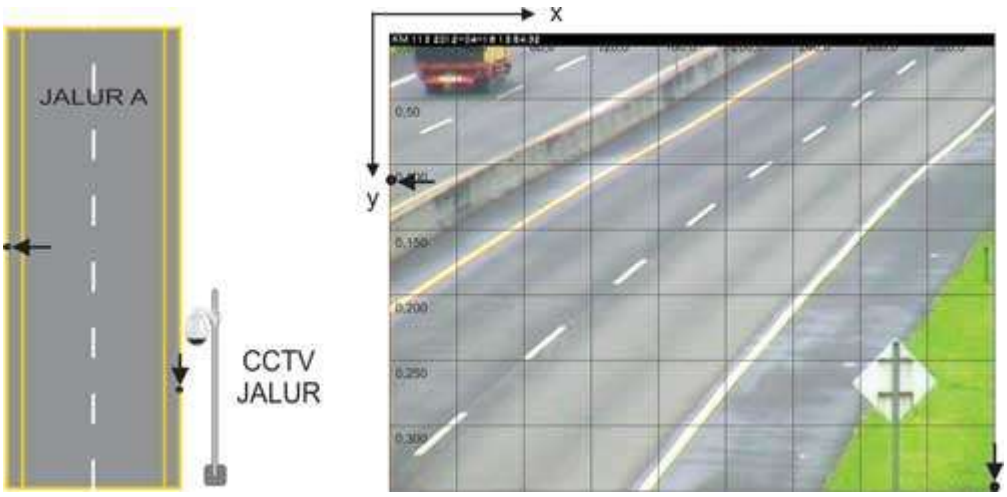


Gambar III.9 Contoh Runtunan Citra

Pada contoh citra di atas, jika objek senantiasa terdapat pada area x=1 (gambar iv) sampai dengan x=8 (gambar i) , serta y=3 (gambar i) sampai

dengan y=8 (gambar iv) , maka titik-titik yang didapatkan adalah (1,3) dan (8,8).

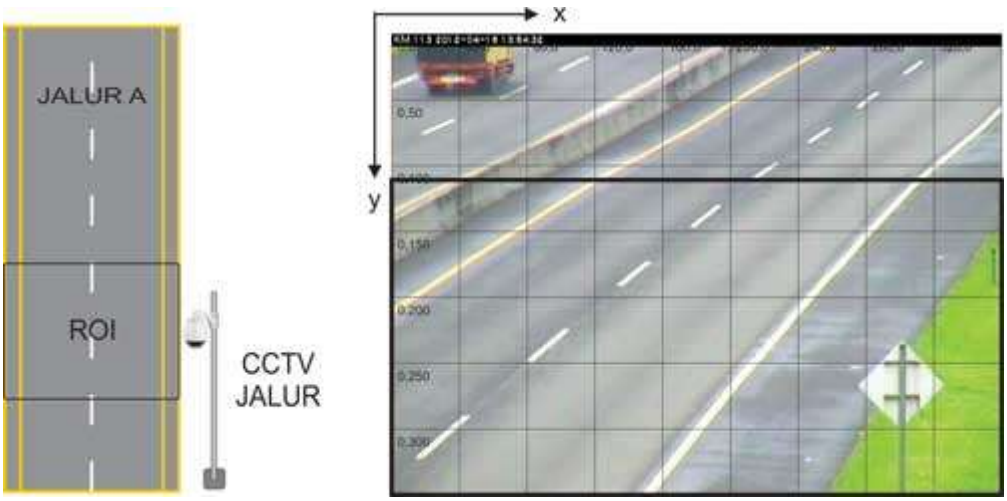
Contoh penentuan titik pada citra yang menjadi objek penelitian dalam tugas akhir ini adalah seperti pada gambar III.10:



Gambar III.10 Penentuan Titik Untuk Pembuatan ROI

C. Membuat gambar persegi panjang

Setelah melakukan proses menetukan titik, proses selanjutnya dalam pembuatan *Region of Interest* (ROI) adalah proses pembutan gambar persegi panjang dengan titik-titik yang telah ditentukan tersebut. Contoh pembuatan persegi panjang dapat dilihat pada gambar III.11:



Gambar III.11 *Region of Interest*

Selain membuat ROI pada background, ROI juga digunakan pada tiap frame video. Posisi ROI pada background dan frame haruslah sama karena di posisi inilah akan dilakukan proses penghitungan NSSD.



Gambar III.12 Contoh Penggunaan ROI Pada Frame

Algoritma untuk menampilkan ROI adalah sebagai berikut:

```
Kamus
   ImageHeight : integer
   ImageWidth : integer
   FrameQty : integer
   roiStartX, roiStartY, roiEndX, roiEndX : integer
   object : Rectangle
   sampleObjects : array[] of Rectangle

Struct Rectangle
   Height, Width : integer
   X, Y : integer (*upper-left coordinate*)

Algoritma
   roiStartX ← ImageWidth
   roiStartY ← ImageHeight
   roiEndX ← 0
   roiEndY ← 0

   Repeat FrameQty
      object ← sampleObjects[FrameQty]

      If (object.X < roiStartX)
         roiStartX ← object.X (*top-left x-coordinate *)
      EndIf

      If (object.Y < roiStartY)
         roiStartY ← object.Y (*top-left y-coordinate *)
      EndIf

      If ((object.X + object.width) > roiEndX)
         roiEndX ← object.X + object.width  (*bottom-right x-coord *)
      EndIf
```

```
      If ((object.Y + object.height) > roiEndY)
         roiEndY ← object.Y + object.height (*bottom-right y-coord *)
      EndIf

      FrameQty ← FrameQty + 1
   Until (FrameQty = 15)

   ShowRoi(roiStartX,roiStartY,roiEndX,roiEndX)
End.
```

Gambar III.13 Algoritma Proses Penentuan ROI

3.4.2.3 *Grayscaling*

Proses selanjutnya setelah proses setting *Region of Interest* adalah proses *Grayscaling*. Proses *Grayscaling* ialah mengubah citra berwarna menjadi citra abu-abu guna mempermudah proses perhitungan. Proses konversi dari citra berwarna menjadi *Grayscale* menggunakan koefisian dari ITU-Recommendation BT.709. Formulasi grayscale untuk satu piksel yang mengandung tiga warna RGB adalah:

$$Grayscale = 0.2125*red + 0.7154*green + 0.0721*blue. \quad (3.1)$$

Gambar III.14 adalah contoh citra berwarna (RGB):



Gambar III.14 Citra Berwarna RGB

Gambar III.15 adalah contoh citra yang telah di *Grayscale*:



Gambar III.15 Citra Grayscale

Sedangkan algoritma grayscale untuk seluruh piksel adalah sebagai berikut:

```
Kamus
   ImageHeight : integer
   ImageWidth : integer
   Image : array[ImageHeight][ ImageWidth] of integer
   RGBImage : array[ImageHeight][ ImageWidth] of integer
   GrayImage : array[ImageHeight][ImageWidth] of integer

Algoritma
   For i ← 0 to ImageWidth do
      For j ← 0 to ImageHeight do
         RGBImage[i][j] ← Image[i][j]
         GrayImage[i][j] ←
             0.2125 * RGBImage[i][j].R +
             0.7154 * RGBImage[i][j].G +
             0.0721 * RGBImage[i][j].B
      Endfor
   Endfor
End.
```

Gambar III.16 AlgoritmaProses *Grayscale*

3.4.2.4 Perhitungan NSSD

Setelah melakukan proses setting ROI dan mengubah citra menjadi citra gray, langkah selanjutnya menghitung nilai *Normalized Sum Squared Differences* (NSSD) yang terdapat pada setiap frame yang telah dikonversi menjadi *grayscale.* Perhitungan dilakukan di dalam ROI. Dalam perhitungan NSSD dibutuhkan nilai intensitas piksel (derajat keabuan) citra *background* dan citra frame. Langkah-langkah Metode NSSD adalah sebagai berikut:

1. Citra *background* dan nilai pikselnya

   Tentukan citra *background* dan nilai pikselnya. Citra ini harus sudah dalam bentuk citra abu-abu. Pada citra bergerak seperti video atau *stream* data,

background tidak ditentukan secara statis atau dengan mengeset satu gambar tertentu, tetapi melalui proses *overlay* frame per frame yang pada akhirnya akan menghasilkan citra *background* yang dinamis.

Untuk mendapatkan citra background dinamis diperlukan FilterObjekBergerak. Langkah-langkah dalam FilterObjekBergerak adalah:

- Hasil dari filter ini adalah sebuah image (result) yang didasarkan pada image sumber (background), yang diupdate dengan cara menurunkan selisih dengan image overlay (frame). Image sumber bergerak menuju image yang di-overlay.
- Persamaan untuk mengupdate image adalah:
  Image Result = Image Source + Min(Abs(Image Overlay – Image Source), StepSize) * Sign(Image Overlay – Image Source)
- Lebih besar nilai StepSize maka akan semakin menghasilkan image yang mirip dengan image overlay.
- StepSize diset dengan nilai 1 sehingga menghasilkan perbedaan yang sangat kecil dengan image sumber (background). Namun, jika filter ini diimplementasikan berulang pada image (result) secara terus-menerus maka akan menjadi sama dengan image overlay dalam maksimum 255 kali iterasi.

Algoritma untuk mendapatkan citra *background* adalah sebagai berikut:

```
Procedure FilterObjekBergerak ( ByRef bground, ByRef frame )

KAMUS
      pixelFormat : PixelFormat ← PixelFormat.Format8bppIndexed
      width : integer ← image.Width
      height : integer ← image.Height
      v, x, y : integer
      stepSize : integer ← 1 (* nilai gradasi pixel *)
      pixelSize : integer  ←  1
      lineSize : integer  ←  width * pixelSize;
      srcOffset : integer  ←  bground.Stride - lineSize;
      ovrOffset : integer  ←  frame.Stride - lineSize;

ALGORITMA

      *bgroundPtr : pointer of byte  ←  bground.ImageData.ToPointer()
      *framePtr : pointer of byte  ←  frame.ImageData.ToPointer()

      (* untuk setiap baris *)
      For y ← 0 to height do
          (* untuk setiap pixel *)
          For x ← 0 to lineSize do
              v = *framePtr - *bgroundPtr;
              (* jika nilai pixel frame lebih besar dari background *)
              If ( v > 0 )
                 (* set background dgn gradasi lebih terang 1 derajat
keabuan *)
                 *bgroundPtr ← *bgroundPtr + stepSize
```

```
                EndIf

                (* jika nilai pixel frame lebih kecil dari background *)
                ElseIf ( v < 0 )
                   v = -v;
                   (* set background dgn gradasi lebih gelap 1 derajat
keabuan *)
                   *bgroundPtr ← *bgroundPtr - stepSize
                EndIf

                x ← x + 1
                bgroundPtr ← bgroundPtr + 1
                framePtr ← framePtr + 1
            EndFor

            bgroundPtr ← bgroundPtr + srcOffset;
            framePtr ← framePtr + ovrOffset;
            y ← y + 1
        EndFor
END
```

Gambar III.17 Algoritma Proses FilterObjekBergerak

Selanjutnya dilakukan proses Difference Filter. Difference Filter berfungsi untuk menentukan selisih antara image background dan frame. Filter ini menggunakan dua image (bakground dan frame) dengan ukuran dan format piksel yang sama dan kemudian membuat image baru, yang mana setiap pixel merupakan selisih absolut dari setiap pixel yang bersesuaian pada kedua image tersebut.

```
Procedure FilterDifference ( ByRef bground, ByRef frame )
{
    pixelFormat : PixelFormat  ←  PixelFormat.Format8bppIndexed
    width : integer  ← frame.Width
    height : integer ← frame.Height
    v, x, y : integer
    pixelSize : integer ← 1
    lineSize : integer  ← width * pixelSize
    srcOffset : integer ← frame.Stride - lineSize
    ovrOffset : integer ← bground.Stride - lineSize

    (* convert ke pointer *)
    *ptr : pointer of byte ← frame.ImageData.ToPointer( )
    *ovr : pointer of byte ← bground.ImageData.ToPointer( )

    (* untuk setiap baris *)
    For y ← 0 to height do
        (* untuk setiap pixel *)
        For x ← 0 to lineSize do
            (* absolute (sub) *)
            v ← *ptr - *ovr
            *ptr ← ( v < 0 ) ? -v : v
            x ← x + 1
        EndFor
        ptr ← ptr + srcOffset
        ovr ← ovr + ovrOffset
        y ← y + 1
     EndFor
END
```

Gambar III.18 Algoritma Proses Filter Diffference

```
Procedure ThresholdPixel (ByRef frame, Rectangle rect )

KAMUS
     threshold : integer  ← 15
     startX  : integer  ← rect.Left
     startY  : integer  ← rect.Top
     stopX   : integer  ← startX + rect.Width
     stopY   : integer  ← startY + rect.Height
     offset  : integer  ← frame.Stride - rect.Width

ALGORITMA

     (* convert ke pointer *)
     *ptr : pointer of byte ← frame.ImageData.ToPointer()

     (* arahkan pointer ke pixel pertama dari RECT untuk diproses *)
     ptr ← ( startY * frame.Stride + startX )

     (* untuk setiap baris *)
     For y ← startY to stopY do
          (* untuk setiap pixel *)
          For x ← startX to stopX do
               (* buat menjadi dua nilai pixel, 255 atau 0 *)
               *ptr ← ( *ptr >= threshold ) ? 255 : 0
                x ← x + 1
                ptr ← ptr + 1
          EndFor
          ptr ← ptr + offset
          y ← y + 1
     EndFor
END
```

Gambar III.19 Algoritma Proses Threshold Pixel Frame

2. Tentukan citra frame dan nilai pikselnya

   Tentukan citra *frame* dan nilai pikselnya. Citra ini juga harus sudah dalam bentuk citra abu-abu. Citra frame didapatkan langsung tanpa melalui pemrosesan lain. Citra frame didapatkan melalui ekstraksi bitmap. Citra frame ini sendiri telah didapatkan dan digunakan untuk mendapatkan citra background di atas.

3. Untuk setiap frame, lakukan pemrosesan setiap piksel yang menggunakan persamaan NSSD berikut:

$$NSSD = \frac{\Sigma \left(P_f(i) - P_b(i) \right)^2}{N}$$

   dengan P_f adalah nilai piksel *frame*, P_b adalah nilai piksel *background*, dan N adalah jumlah piksel yang diproses.

4. Tentukan nilai rata-rata NSSD.

3.4.2.5 *Threshold*

Tahapan selanjutnya adalah *thresholding* atau penentuan nilai ambang batas untuk memastikan apakah runtunan citra frame merupakan objek yang bergerak atau bukan. Pada tugas akhir ini proses penentuan nilai ambang batas didapatkan melalui proses *trial and error* pada objek bergerak. Proses ini mendapatkan nilai ambang batas:

1. Nilai kuadrat selisih (*square difference*) > 100

   Piksel dalam citra frame yang memiliki nilai kuadrat selisih yang lebih besar dari 100 dapat dipastikan merupakan piksel objek. Sebaliknya nilai yang lebih kecil dari 100 merupakan bagian dari *background* atau bias.
2. Nilai NSSD > 800

   Kumpulan piksel dalam frame yang memiliki nilai rata-rata NSSD lebih 800 merupakan sebuah objek yang bergerak.

3.4.2.6 Identifikasi Bentuk Objek

Sebelum mendapatkan jumlah kendaraan, perlu dibuat representasi objek kendaraan dalam bentuk persegi untuk mempermudah identifikasi kendaraan. Proses penggambaran bentuk persegi sebuah objek bergerak adalah sebagai berikut:

1. Identifikasi bentuk persegi objek

   Setiap citra *grayscale* dalam frame memiliki piksel-piksel dengan koordinat (x,y) masing-masing. Identifikasi bentuk persegi dilakukan dengan mencari titik terluar dari setiap objek yang terdapat di dalam citra frame.
2. Penggambaran objek sebagai persegi

   Setelah bentuk persegi didapat maka proses selanjutnya yaitu proses penggambaran persegi pada obyek yang terdeteksi.

Contoh identifikasi bentuk objek adalah seperti diperlihatkan pada gambar III.20



Gambar III.20 Bentuk Persegi dari Sebuah Objek

Algoritma Identifikasi Bentuk Objek diperlihatkan pada gambar III.21:

```
Procedure IdentifikasiObjekDalamFrame( ByRef frame )

KAMUS
   backgroundThresholdR : byte = 15
   backgroundThresholdG : byte = 15
   backgroundThresholdB : byte = 15

   stride : integer = image.Stride;
   imageWidthM1 : integer = imageWidth - 1;
   objectLabels : array of integer [imageWidth * imageHeight]
   labelsCount : integer = 0;
   i : integer
   p : integer = 0
   offset : integer = stride - imageWidth
   maxObjects : integer = ((imageWidth/2) + 1) * (( imageHeight / 2 ) + 1) + 1
   map : array of integer = new int[maxObjects];

   *src : pointer of byte = image.ImageData.ToPointer( )

   (* awalnya, petakan semua label dengan dirinya sendiri *)
   For i ← 0 to maxObjects do
       map[i] = i;
   EndFor

   (* 1 - untuk pixel di baris pertama *)
   If ( *src > backgroundThresholdG )
      objectLabels[p] = ++labelsCount;
   EndFor

   ++src;
   ++p;

   (* proses sisa pixel di baris pertama *)
   For ( int x = 1; x < imageWidth; x++, src++, p++ )
      (* cek apakah diperlukan label untuk current pixel *)
      If ( *src > backgroundThresholdG )
         (* cek apakah pixel sebelumnya sudah dilabel atau belum *)
         If ( src[-1] > backgroundThresholdG )
            (* labeli current pixel seperti sebelumnya *)
            objectLabels[p] = objectLabels[p - 1]
         Else
            (* buat label baru *)
            objectLabels[p] = ++labelsCount
         EndIf
```

```
      }
   }
   src += offset;
   (* 2 - untuk baris lainnya *)
   (* untuk setiap baris *)
   For ( int y = 1; y < imageHeight; y++ )

      (* untuk pixel pertama dari setiap baris, *)
      (* hanya perlu mengecek pixel kanan teratas *)
      If ( *src > backgroundThresholdG )
         (* cek pixel di sekitar *)
         If ( src[-stride] > backgroundThresholdG )
            objectLabels[p] = objectLabels[p - imageWidth];
         ElseIf ( src[1 - stride] > backgroundThresholdG )
            objectLabels[p] = objectLabels[p + 1 - imageWidth];
         Else
            objectLabels[p] = ++labelsCount;
         EndIf
      EndIf
      ++src;
      ++p;

      (* cek pixel kiri dan tiga pixel sebelah atas dari *)
      (* semua pixel yang tersisa *)
      For ( int x = 1; x < imageWidthM1; x++, src++, p++ )
         If ( *src > backgroundThresholdG )
            (* cek pixel di sekitar *)
            if ( src[-1] > backgroundThresholdG )
               objectLabels[p] = objectLabels[p - 1];
            EndIf

            (* label objek jika belum dilabeli *)
            If ( objectLabels[p] == 0 )
               objectLabels[p] = ++labelsCount;
            EndIf
         EndIf
      EndFor

      (* untuk semua pixel di baris ini, hanya perlu mengecek pixel *)
      (* sebelah atas dan sebelah kiri atas *)
      If ( *src > backgroundThresholdG )
         if ( src[-1] > backgroundThresholdG )
            objectLabels[p] = objectLabels[p - 1];
         Else
            objectLabels[p] = ++labelsCount;
         EndIf
      EndIf
      ++src;
      ++p;
      src += offset;
   EndFor
```

Gambar III.21 Algoritma Identifikasi Bentuk Objek

Algoritma untuk mendapatkan objek dalam bentuk rectangle atau persegi diperlihatkan pada gambar III.22:

```
Function GetObjectsRectangles ( UnmanagedImage frame, Rectangle rect ) : array of
Rectangle
     i : integer
    rects : array Rectangle ← Rectangle[objectsCount]
    For i ← 0 to objectsCount do
        rects[i] ← blobs[i].Rectangle
        i ← i + 1
    EndFor

    return rects
END
```

Gambar III.22 Algoritma GetObjectsRectangles

3.4.2.7 Penghitungan Kendaraan dan Tingkat Kepadatan

Proses penghitungan kendaraan melalui beberapa proses yang diuraikan sebagai berikut:

1. Penentuan luas objek

   Objek akan dipilah berdasarkan ukuran atau jumlah piksel dalam objek yang sudah berbentuk persegi. Jumlah piksel ini bergantung kepada faktor posisi kamera, baik ketinggian maupun kemiringan terhadap ROI. Jika luas objek memenuhi kriteria, maka objek layak untuk diproses lebih lanjut. Ketentuan luas objek yang merupakan kendaraan adalah sebagai berikut:

   - Luas minimum adalah 17.000 piksel



Gambar III.23 Luas Minimum Objek Kendaraan

   Luas minimum ini didapatkan dengan menghitung jumlah piksel mobil terkecil (kelas sedan dan sejenisnya), yaitu 17.579.

   - Luas maksimum adalah 75.000 piksel.



Gambar III.24 Luas Maksimum Objek Kendaraan

Jika jumlah piksel melampaui 75.000 maka dipastikan objek merupakan gabungan dari beberapa objek sekaligus, bukan sebagai satu objek tunggal. Nilai 75.000 didapatkan setelah mengamati sebuah kendaraan truk terpanjang yang melintas. Jika jumlah piksel lebih dari 75.000, maka harus dilakukan penghitungan NSSD lagi untuk seri frame berikutnya sehingga benar-benar didapatkan satu objek tunggal.

2. Penambahan Jumlah Kendaraan

   Jika objek terdepan melewati atau meninggalkan ROI, maka jumlah kendaraan dinaikkan satu.

3. Penentuan Tingkat Kepadatan Kendaraan

Tingkat kepadatan ditentukan berdasarkan konstanta kepadatan yang telah diberikan oleh pihak yang berkepentingan. Jika jumlah kendaraan yang terhitung pada satu titik pantau memenuhi kriteria tingkat kepadatan, maka informasi tingkat kepadatan akan disesuaikan atau ditampilkan ke user. Nilai konstanta tingkat kepadatan ditentukan oleh pihak pengelola jalan atau instansi terkait, dalam hal ini PT Jasa Marga Tbk.

Algoritma perhitungan kendaraan dan tingkat kepadatan diuraikan pada Gambar III.29 berikut:

```
Procedure HitungKendaraan(Bitmap image)

Kamus
   imageHeight, imageWidth : integer
   candidate, movingObject : Rectangle
   bitmapData : BitmapData
   blobCounter : BlobCounter
   objectsCandidate : array[] of Rectangle
   idx, counter, trafficDensity : integer
   objectGoingToLeft, isLeaveRoi, isCounted : boolean

Struct Rectangle
   Height, Width : integer
   X, Y : integer (*upper-left coordinate*)

Algoritma
   counter ← 0
   objectsCandidate ← GetObjectRectangles(bitmapData)
   If (objectsCandidate.Length <> 0)
      Repeat idx
         candidate = objectsCandidate[idx]
         If (isKendaraan(candidate.Height * candidate.Width)
            If (objectGoingToWest = TRUE)
               If (candidate.Y > roiStartY)
                  If (candidate.X < movingObject.X)
                     movingObject = candidate
                  EndIf
               EndIf
            EndIf
         EndIf
      Until (idx = objectsCandidate.Length)
```

```
      If (objectGoingToLeft = TRUE)
         If (movingObject.X = roiStartX)
            isLeaveRoi ← TRUE
            isCounted ← TRUE
            movingObject ← NULL
         ElseIf (movingObject.X > roiStartX)
            isCounted ← FALSE
         EndIf
      EndIf
      If (isLeaveRoi = TRUE And isCounted = FALSE)
         counter ← counter + 1
         isLeaveRoi ← TRUE
         isCounted ← FALSE
         addLalin()
      EndIf
   EndIf
End.

Function isKendaraan(pixel: integer) : boolean

Algoritma
   if (pixel > 17000 And pixel < 75000)
      return true
   else
      return false
End.

Procedure addLalin()

Kamus
   Query: array[] of character
   Lalin l Lalin()
   sqlConnection MySqlConnection()

Algoritma
   Query = "INSERT INTO TblLalin(IdCabang, IdPetugas, IdCamera, Tanggal, Panjang,
Lebar, Luas, Status) VALUES ( l.IdCabang, l.IdPetugas, l.IdCamera,
'l.EventDate.ToString("yyyy-MM-dd HH:mm:ss")', l.Vhcl.Height,
l.Vhcl.Width,(l.Vhcl.Height * l.Vhcl.Width), l.Status)
   MySqlCommand(Query, sqlConnection)
End.
```

Gambar III.25 Algoritma penghitungan kendaraan

```
Function VehicleDensity (numOfVehiclePerTimeUnit : integer) : array[] of
character

Kamus
   vhclDensity : array[] of character

Algoritma

   If (numOfVehiclePerTimeUnit <= 1400)
         vhclDensity= "Sepi"
   ElseIf (numOfVehiclePerTimeUnit > 1400 And numOfVehiclePerTimeUnit <= 2275)
         vhclDensity = "Sedang"
   ElseIf (numOfVehiclePerTimeUnit >= 2275 And numOfVehiclePerTimeUnit < 2975)
         vhclDensity = "Padat"
   Else
         vhclDensity = "Macet"
   EndIf
   Return vhclDensity
End.
```

Gambar III.26 Algoritma tingkat kepadatan

Secara keseluruhan Algoritma Sistem Pemantau Kepadatan Lalu Lintas Kendaraan Otomatis, diuraikan pada gambar III.31 sebagai berikut:

```
Kamus
   height, width, numOfFrame, n, i, j : integer
   Xstart, Ystart, Xend, Yend : integer
   URL : string
   stream : MJPEGStream
   sourceImage : pointer of pointer of character
   bitmapRGB : array[][] of integer
   bitmapGray : array[][] of integer
   background : array[][] of integer
   frameProcess : array[][] of integer
   candidateObject : array[][] of integer
   nssdValue : real
   numObject : integer

   constant squareDifference : integer = 100
   constant nssdReference : integer = 800
   constant sepi : integer = 1400
   constant sedang : integer = 2275
   constant padat : integer = 2975
   constant macet : integer = 3500

   constant areaMin : integer = 17500
   constant areaMax : integer = 75000

procedure VideoCapture (img : MJPEGStream)

function ConvertToBitmapRGB(frame : array[][] of integer)
       -> array[][] of integer

function ConvertToBitmapGray(rgb : array[][] of integer)
       -> array[][] of integer

procedure SetRoi(gray : array[][] of integer, Xstart,
          Ystart, Xend, Yend)

procedure SetBackgound(gray : array[][] of integer)

function SquareDifference(bground : array[][] of integer,
         frameProcess : array[][] of integer )
       -> integer

function ProcessNSSD(bground : array[][] of integer,
         frameProcess : array[][] of integer, Ref candidateObject)
       -> integer

function GenerateBentukObjek(object : array[][] of integer)
       -> array[][] of integer

function VehicleDensity(numOfVehiclePerTimeUnit : integer) : array[] of character

Algoritma
   URL ← "http://202.90.195.131:218/axis-cgi/
            mjpg/video.cgi"
   stream.VideoSource ← URL
   stream.Start()
   if (stream <> NULL) then
      sourceImage ← stream.GetSourceImage()
      numOfFrame ←  stream.NumOfFrame
      height ← sourceImage.Height
      width ← sourceImage.Width
      n ← 0
      repeat numOfFrame
         bitmapRGB[height][width] ←
            ConvertToBitmapRGB(sourceImage)
         bitmapGray[height][width] ←
            ConvertToBitmapGray(bitmapRGB)
         ShowRoi(bitmapGray, Xstart, Ystart, Xend, Yend)
         background ← GetBackgound(bitmapGray)
         squareDifference ←
```

```
SquareDifference(background,bitmapGray,candidateObject)
         nssdValue ← ProcessNSSD(background,bitmapGray,squareDifference)

         n ← n + 1
      until (n = NumOfFrame)
      If (nssdValue > 800)
         numObject ← numObject + 1
         GenerateBentukObjek(candidateObject)
         VehicleDensity(numOfVehiclePerTimeUnit)
      EndIf
   EndIf
   stream.Stop()
```

Gambar III.27 Algoritma Sistem Pemantau Kepadatan Lalu Lintas Kendaraan Otomatis

3.4.3 Analisis Kebutuhan Non Fungsional

Analisis kebutuhan non fungsional merupakan analisis yang dibutuhkan untuk menentukan spesifikasi kebutuhan sistem. Spesifikasi ini juga meliputi elemen atau komponen-komponen apa saja yang dibutuhkan untuk sistem yang akan dibangun sampai dengan sistem tersebut ` dapat di implementasikan.

Pada Analisis non fungsional terdapat 3 kriteria utama, yaitu kondisi *existing* sistem yang sedang berjalan, usulan untuk sistem yang direkomendasikan kepada pengguna, serta solusi yang ditawarkan agar sistem yang dibangun dapat mudah diterima oleh pengguna(*user friendly*) dan perangkat keras yang digunakan dapat mendukung berjalannya sistem secara maksimal terhadap kinerja sistem.

Analisis kebutuhan ini juga menentukan spesifikasi masukan yang diperlukan sistem, keluaran yang akan dihasilkan dan proses yang dibutuhkan untuk mengolah masukan sehingga menghasilkan suatu keluaran yang diinginkan.

3.4.3.1 Analisis Pengguna

Analisis pengguna dilakukan untuk mengetahui siapa yang akan menggunakan aplikasi ini antara lain:

A. Pelaku sistem yang sedang berjalan

Berdasarkan hasil observasi dan penelitian yang dilakukan di bagian pelayanan lalu lintas PT.Jasa Marga(Persero), tugas dari pelaku pada sistem yang sedang berjalan adalah sebagai berikut:

Tabel III.1 Pelaku Sistem yang sedang berjalan

Pengguna	Tugas	Keterangan
Petugas Sentra Komunikasi	- Memonitor dan mengendalikan kamera CCTV. - Menerima dan meneruskan laporan dari lapangan. - Memprediksi kondisi lalu lintas. - Membuat laporan yang ditulis manual pada form yang telah disediakan.	Petugas pemantau lalu lintas yang berada di cabang Jasa Marga (Purbaleunyi)
Unit Sarana Elektronik	Memonitor kondisi perangkat pemantauan agar tetap terjaga dan berfungsi dengan baik.	Unit pemeliharaan dan perbaikan perangkat pemantauan.
JMTIC (Jasa Marga Trafic Information System)	Memonitor kamera CCTV	Pusat Informasi yang berada di Kantor Pusat Jasa Marga.
Kepolisian	Memonitor kamera CCTV	Instansi yang bekerja sama dengan Jasa Marga dalam pemantauan lalu lintas di jalan tol
Kasubag YANLALIN	Memeriksa laporan kepadatan lalu lintas dari petugas sentra komunikasi	Atasan dari petugas sentra komnunikasi

B. Pengguna sistem yang baru

Untuk penerapan sistem yang baru, tugas dari petugas sentra komunikasi akan dibantu oleh aplikasi yang akan dibangun. Pemantauan kepadatan lalu lintas akan secara otomatis dilakukan oleh sistem, hasil dari pemantauan tersebut akan terdata kemudian data-data tersebut akan disimpan di *database*.

Tabel III.2 Pengguna Sistem Baru

Pengguna	Tugas	Hak Akses
Petugas Sentra Komunikasi	- Memonitor dan mengendalikan kamera CCTV. - Menerima dan meneruskan laporan dari lapangan. - Membuat laporan kepadatan lalu lintas	- Mendapatkan informasi kepadatan lalu lintas - Membuat laporan kepadatan lalu lintas

	dari aplikasi pemantauan kepadatan lalu lintas.	

C. GAP Analisis Pengguna Sistem

GAP Analisis Pengguna Sistem adalah pada pemantauan dan pembuatan laporan. Pada sistem yang lama pemantauan dan pelaporan dilakukan secara manual oleh petugas sentra komunikasi sedangkan pada sistem yang baru pemantauan kepadatan lalu lintas akan secara otomatis dilakukan oleh sistem, hasil dari pemantauan tersebut akan terdata kemudian data-data tersebut akan disimpan di *database*, kemudian data-data tersebut yang ada di *database* diolah untuk dibuat laporan kepadatan lalu lintas

3.4.3.2 Analisis Kebutuhan Perangkat Keras

Perangkat keras untuk implementasi program perangkat lunak sangatlah mendukung agar menghasilkan suatu informasi yang sesuai dengan yang diharapkan.

A. Spesifikasi yang digunakan pada sistem yang sedang berjalan

Berdasarkan hasil observasi dan penelitian langsung terhadap pihak PT.Jasa Marga(persero) Cabang Purbaleunyi, didapati perangkat keras yang digunakan dalam pemantauan kepadatan lalu lintas adalah sebagai berikut:

Tabel III.3 Spesifikasi perangkat keras sistem yang sedang berjalan

Infrastruktur	Nama Perangkat Keras	Deskripsi
Jalur	Kamera CCTV	Kamera AXIS
Server	CPU	IBM type X3650
	Monitor	LCD Monitor 17”
	Harddisk	10 TB
Client	CPU	P4 1.8 Ghz
	RAM	512 MB
	Monitor	LCD Monitor 42”
	Harddisk	20 GB
Perangkat keras pendukung komunikasi data	Kabel FO	
	OTB Box	
	Converter FO	
	Kabel UTP/STP	
	Switch HUB	
	Switch HUB FO	

B. Kebutuhan perangkat keras sistem yang akan dibangun

Kebutuhan perangkat keras sistem yang akan dibangun adanya penambahan server yang berfungsi sebagai terminal untuk aplikasi pemantauan kepadatan lalu lintas, Spesifikasi dari perangkat keras dijelaskan dalam tabel berikut:

Tabel III.4 Spesifikasi perangkat keras sistem yang akan dibangun

Infrastruktur	Nama Perangkat Keras	Deskripsi
Jalur	Kamera CCTV	Kamera AXIS
Server	CPU	IBM type X3650
	Monitor	LCD Monitor 17”
	Harddisk	10 TB
Client	CPU	P4 1.8 Ghz
	RAM	512 MB
	Monitor	LCD Monitor 42”
	Harddisk	20 GB
Perangkat keras pendukung komunikasi data	Kabel FO	
	OTB Box	
	Converter FO	
	Kabel UTP/STP	
	Switch HUB	
	Switch HUB FO	

C. GAP Analisis Kebutuhan Perangkat Keras

Agar sistem yang dikembangkan dapat berjalan dengan maksimal, maka disarankan untuk dapat memenuhi kriteria kebutuhan perangkat keras untuk sistem yang baru.

3.4.3.3 Analisis Kebutuhan Perangkat Lunak

Kebutuhan perangkat lunak merupakan faktor yang harus dipenuhi untuk merancang sebuah perangkat lunak sehingga sesuai dengan maksud dan tujuan perangkat lunak yang akan dibangun.

A. Perangkat lunak pada sistem yang Berjalan Saat Ini

Berdasarkan hasil observasi dan penelitian langsung terhadap pihak PT.Jasa Marga(persero) Cabang Purbaleunyi, didapati perangkat keras yang digunakan dalam pemantauan kepadatan lalu lintas diperlihatkan pada tabel III.5:

Tabel III.5 Spesifikasi Perangkat lunak pada sistem yang Berjalan Saat Ini

Infrastruktur	Nama Perangkat Lunak	Deskripsi
Server	Sistem Operasi	Windows Server 2008 R2
	AXIS OCS Server	Aplikasi Recording video kamera CCTV
Client	Sistem operasi	Windows 7
	AXIS OCS Client	Aplikasi untuk menampilkan video kamera CCTV
Pendukung komunikasi data		Jaringan LAN (Local Area Network)

B. Kebutuhan Perangkat Lunak Sistem Baru

Kebutuhan perangkat lunak sistem yang akan dibangun adanya penambahan aplikasi di server(terminal pemantau kepadatan lalu lintas), Spesifikasi dari perangkat lunak dijelaskan dalam tabel berikut:

Tabel III.6 Spesifikasi Perangkat lunak pada sistem yang akan dibangun

Infrastruktur	Nama Perangkat Lunak	Deskripsi
Server	Sistem Operasi	Windows Server 2008 R2
	AXIS OCS Server	Aplikasi Recording video kamera CCTV
	IDE	Visual Studio 2008(C#)
	Framework Image Processing	AForge.Net Framework 2.2.4
	DBMS	MySql 5.5.8
	SQL Connector	MySQL Connector Net 6.3.8
Client	Sistem operasi	Windows 7
	AXIS OCS Client	Aplikasi untuk menampilkan video kamera CCTV
	Framework	Framework .Net 3.5
Pendukung komunikasi data		Jaringan LAN (Local Area Network)

C. GAP Analisis Kebutuhan Perangkat Lunak

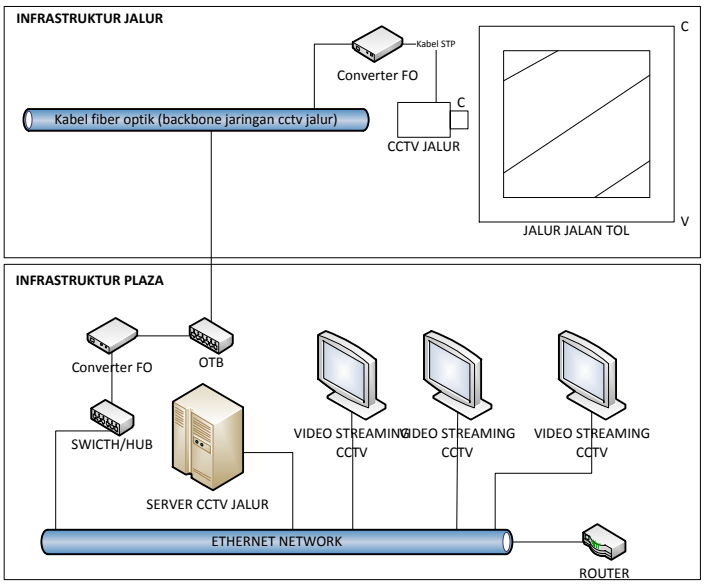
Agar sistem yang dikembangkan dapat berjalan dengan maksimal, maka harus dapat memenuhi kebutuhan perangkat lunak seperti yang telah diuraikan pada kebutuhan perangkat lunak.

3.4.3.4 Analisis Kebutuhan Komunikasi

Kebutuhan komunikasi merupakan faktor yang harus dipenuhi untuk merancang sebuah sistem dengan arsitektur client server, sehingga sesuai dengan maksud dan tujuan yang akan dibangun.

A. Topologi jaringan komputer pada sistem yang berjalan saat ini

Berdasarkan hasil observasi dan penelitian langsung terhadap pihak PT.Jasa Marga(persero) Cabang Purbaleunyi, didapati topologi jaringan komputer yang digunakan dalam pemantauan kepadatan lalu lintas diperlihatkan gambar III.28:



Gambar III.28 Topologi Jaringan pada sistem yang berjalan

B. Topologi jaringan komputer pada sistem baru

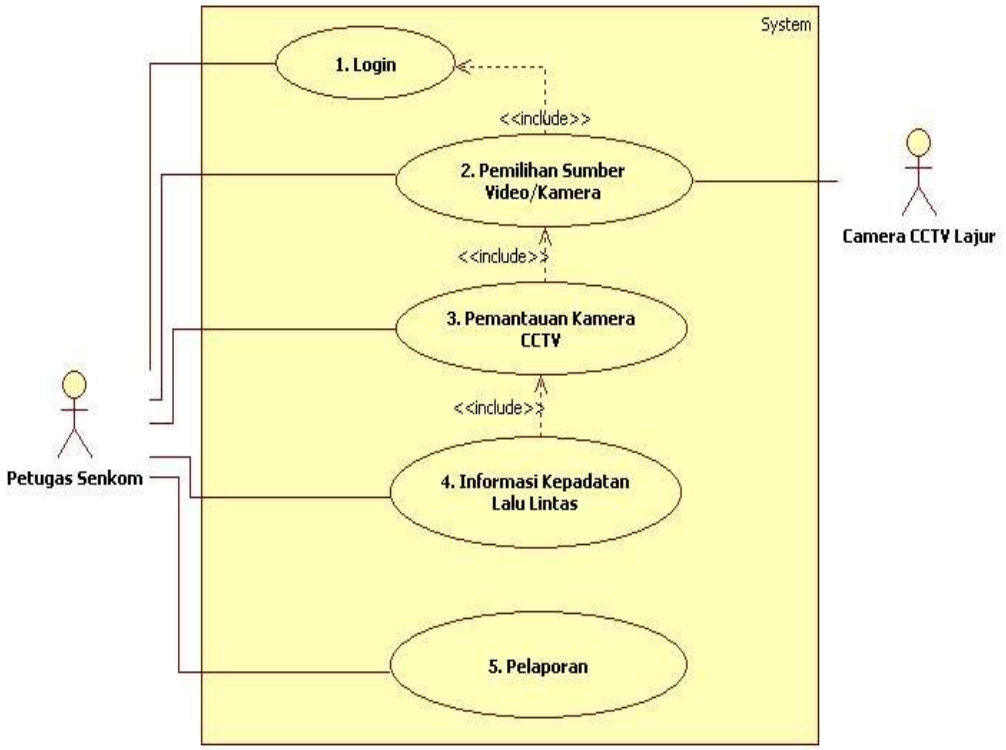
Topologi jaringan komputer pada sistem baru sama seperti pada Topologi jaringan komputer pada sistem yang berjalan.

3.4.4 Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional menjelaskan proses-proses yang terjadi di dalam aplikasi ini. Proses-proses tersebut dijelaskan dalam *usecase diagram, activity diagram, sequence diagram dan Class diagram.*

3.4.4.1 *Use Case Diagram*

Use case diagram menggambarkan proses yang ada di dalam aplikasi ini. Use case diagram aplikasi ini digambarkan pada Gambar III.29:



Gambar III.29 Use Case Sistem Pemantauan Lalu Lintas Otomatis

3.4.4.2 Skenario Diagram

Skenario diagram menjelaskan skenario dari setiap proses yang terdapat pada use case diagram. Skenario diagram pada aplikasi ini dijelaskan pada Tabel III.7 sampai III-11 berikut ini:

Tabel III.7 Skenario diagram proses Login

Identifikasi	
Nomor	1
Nama	Login
Tujuan	Autentikasi user yang bisa menggunakan aplikasi

Deskripsi	Proses autentikasi user
Aktor	Petugas Senkom
Skenario Utama	
Kondisi Awal	Isian textbox id petugas dan password masih kosong
Aksi Aktor	Reaksi Sistem
1. Petugas Senkom berada pada tampilan form login aplikasi.	
2. Petugas Senkom mengisi isian id petugas.	3. Muncul nama petugas di textbox nama petugas sesuai dengan id petugas.
4. Petugas Senkom mengisi isian password	
5. Petugas Senkom memilih tombol login	6. Melakukan validasi data petugas
	7. Jika data salah muncul pesan kesalahan, kemudian kembali ke tahap 4
	8. Jika data login benar, selanjutnya ke proses pemilihan sumber video/kamera.

Tabel III.8 Skenario diagram proses Pemilihan Sumber Video/Kamera

Identifikasi	
Nomor	2
Nama	Pemilihan Sumber Video/Kamera
Tujuan	Pemilihan asal sumber video oleh petugas
Deskripsi	Pilihan sumber video: *video recording*, lokal kamera, *network* kamera
Aktor	Petugas Senkom, Camera CCTV Lajur
Skenario Utama	
Kondisi Awal	Tampilan video pada form utama aplikasi masih kosong
Aksi Aktor	Reaksi Sistem
1. Petugas Senkom berada pada tampilan form utama aplikasi.	
2. Petugas Senkom memilih menu file dan memilih sumber video	3. Muncul tampilan pilihan sumber video
4. Petugas Senkom memilih *open video*	
5. Camera CCTV Lajur memberikan data streaming video.	6. Streaming video tampil di aplikasi berdasarkan pilihan sumber video, selanjutnya ke proses pemantauan kamera cctv.

Tabel III.9 Skenario diagram proses Pemantauan Kamera CCTV

Identifikasi	
Nomor	3
Nama	Pemantauan Kamera CCTV
Tujuan	Pengolahan citra pada streaming video cctv
Deskripsi	Pemantauan secara langsung *streaming video* dan tampilan pengolahan citra pada tampilan utama aplikasi
Aktor	Petugas Senkom
Skenario Utama	
Kondisi Awal	Tampilan *streaming video* pada form utama aplikasi
Aksi Aktor	Reaksi Sistem
1. Petugas Senkom berada pada tampilan form utama aplikasi.	
2. Petugas Senkom melakukan setting koordinat region of interest, ROI(start) dan ROI(end)	3. Perubahan koordinat region of interest.
4. Petugas Senkom melakukan setting minimum dan maksimum *object size*	5. Perubahan luas identifikasi minimum dan maksimum objek yang akan dihitung
6. Petugas Senkom melakukan setting *density level*	7. Perubahan level tingkat kepadatan.
8. Petugas Senkom melakukan cek show Rectangle	9. Tampilan kotak identifikasi calon objek
10. Petugas Senkom melakukan cek coordinate	11. Tampilan koordinat identifikasi calon objek
12. Petugas Senkom melakukan cek show area, cek show roi	13. Tampilan koordinat identifikasi calon objek
14. Petugas Senkom melakukan cek show roi	15. Tampilan kotak region of interest

Tabel III.10 Skenario diagram proses Informasi Kepadatan Lalu Lintas

Identifikasi	
Nomor	4
Nama	Informasi Kepadatan Lalu Lintas
Tujuan	Untuk mengetahui Informasi Kepadatan Lalu Lintas
Deskripsi	Informasi jumlah kendaraan per jam dan tingkat

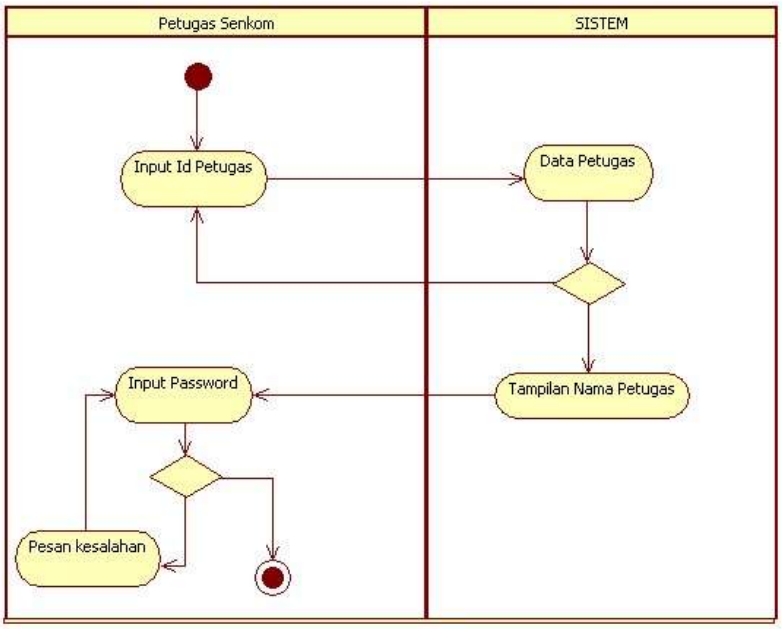
	kepadatan lalu lintas
Aktor	Petugas Senkom
Skenario Utama	
Kondisi Awal	Tampilan *counter* jumlah kendaraan 0 dan tingkat kepadatan sepi
Aksi Aktor	Reaksi Sistem
1. Petugas Senkom berada pada tampilan form utama aplikasi.	
2. Petugas Senkom mengamati jumlah kendaraan per jam dan tingkat kepadatan lalu lintas	3. Jumlah kendaraan bertambah sesuai dengan pendeteksian objek
	4. Tingkat kepadatan berubah berdasarkan penghitungan jumlah kendaraan per jam

Tabel III.11 Skenario diagram proses Pelaporan

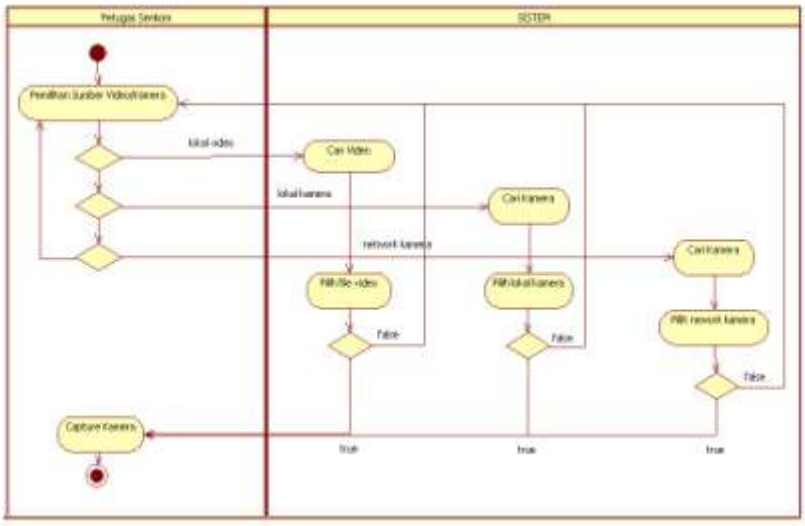
Identifikasi	
Nomor	5
Nama	Pelaporan
Tujuan	Membuat laporan lalu lintas kendaraan
Deskripsi	laporan lalu lintas kendaraan per jam dan laporan lalu lintas kendaraan per hari
Aktor	Petugas Senkom
Skenario Utama	
Kondisi Awal	Tampilan form utama
Aksi Aktor	Reaksi Sistem
1. Petugas Senkom berada pada tampilan form utama aplikasi.	
2. Petugas Senkom memilih menu report	3. Muncul tampilan pilihan submenu report per jam dan report per hari
4. Petugas Senkom memilih menu report per jam	5. Muncul tampilan report viewer
6. Petugas Senkom tanggal dan jam	7. Muncul tampilan laporan berdasarkan pemilihan tanggal dan jam.
8. Petugas Senkom memilih menu report hari	9. Muncul tampilan report viewer
10. Petugas Senkom tanggal	11. Muncul tampilan laporan berdasarkan pemilihan tanggal.

3.4.4.3 *Activity Diagram*

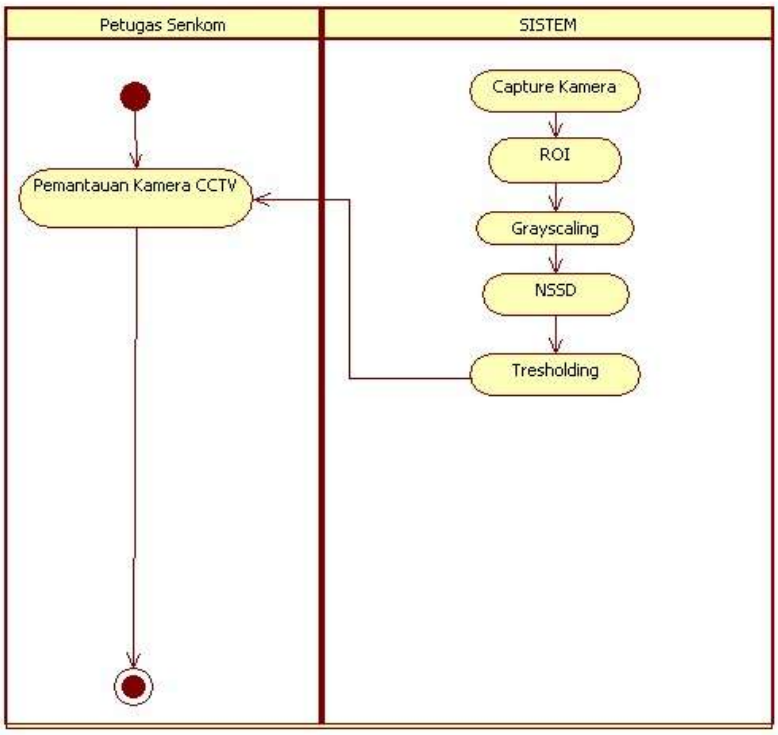
Activity diagram menjelaskan alur aktivitas dari setiap usecase. *Activity diagram* sistem ini dapat dilihat pada gambar III.30 sampai dengan gambar III.34:



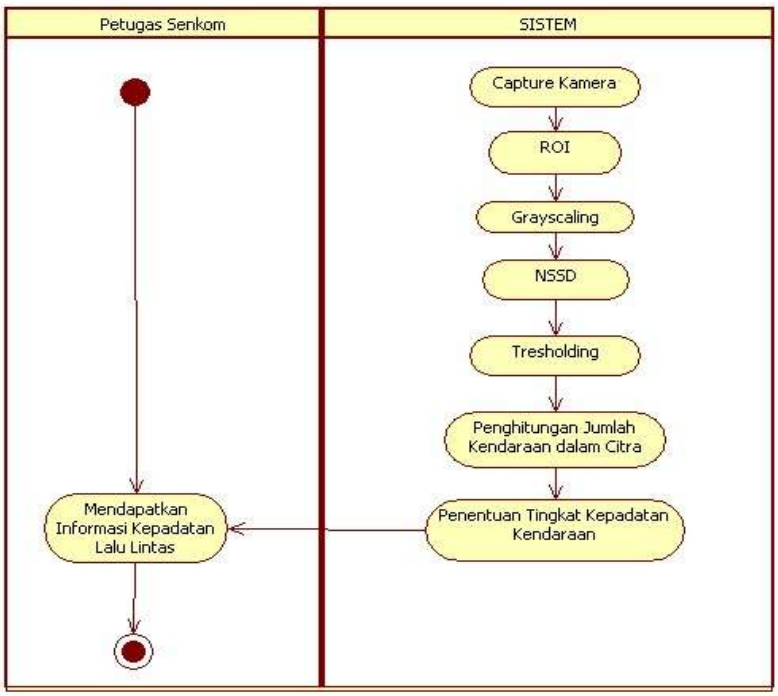
Gambar III.30 *Activity diagram* Login



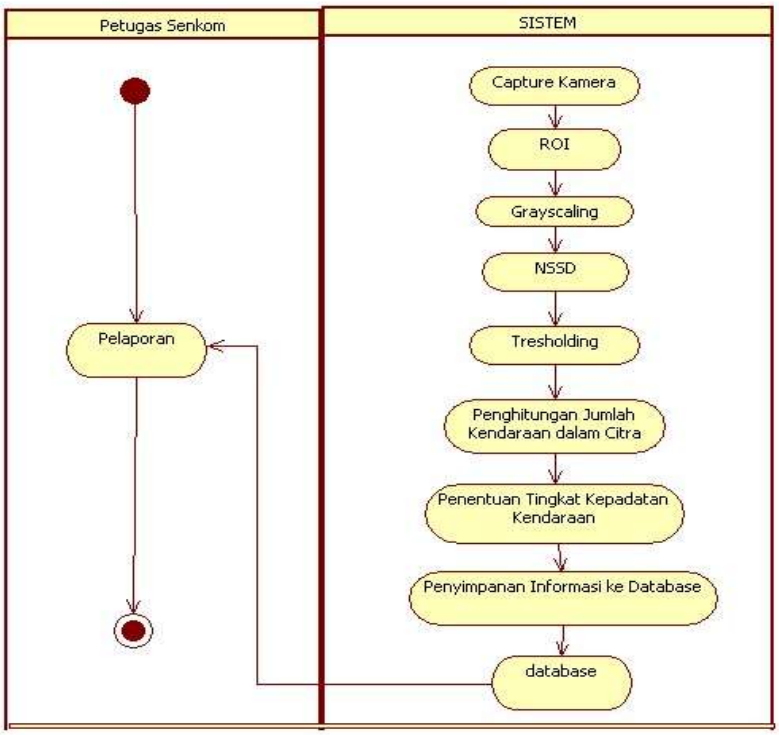
Gambar III.31 *Activity diagram* Pemilihan Sumber Video/Kamera



Gambar III.32 *Activity diagram* Pemantauan Kamera CCTV



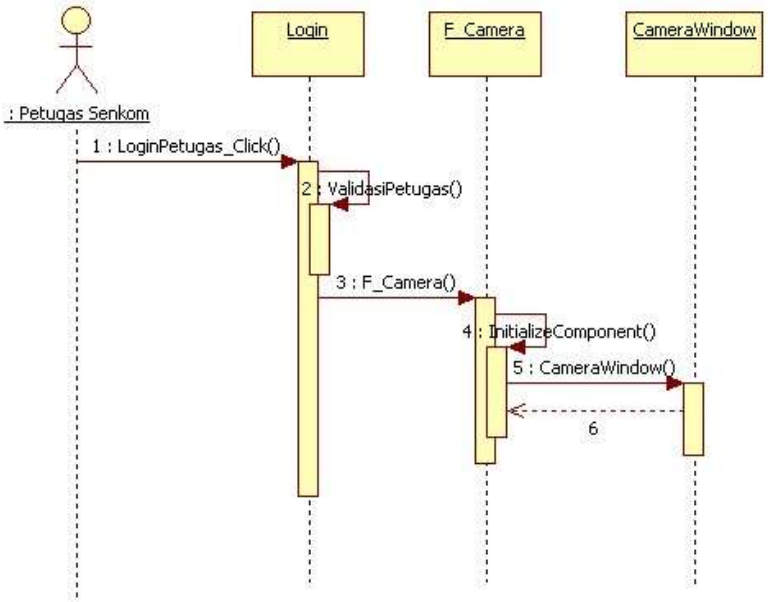
Gambar III.33 *Activity diagram* Mendapatkan Informasi Kepadatan Lalu Lintas



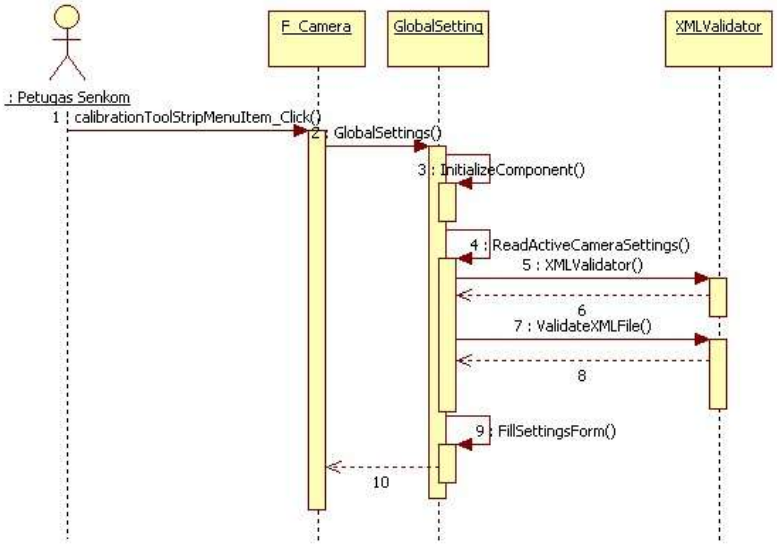
Gambar III.34 *Activity diagram* Pelaporan

3.4.4.3 *Sequence Diagram*

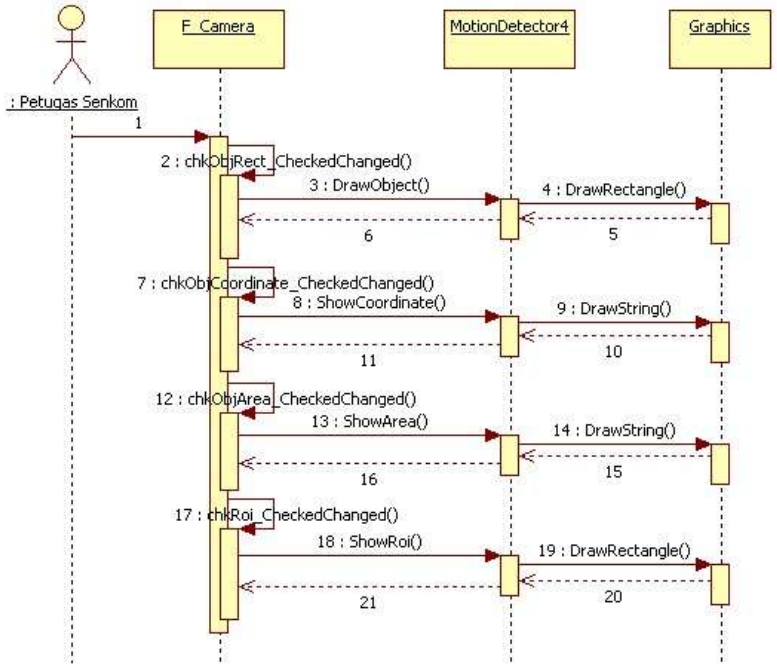
Sequence diagram menjelaskan alur yang akan berjalan dalam sistem ini. Sequence diagram sistem ini ditunjukkan pada Gambar berikut:



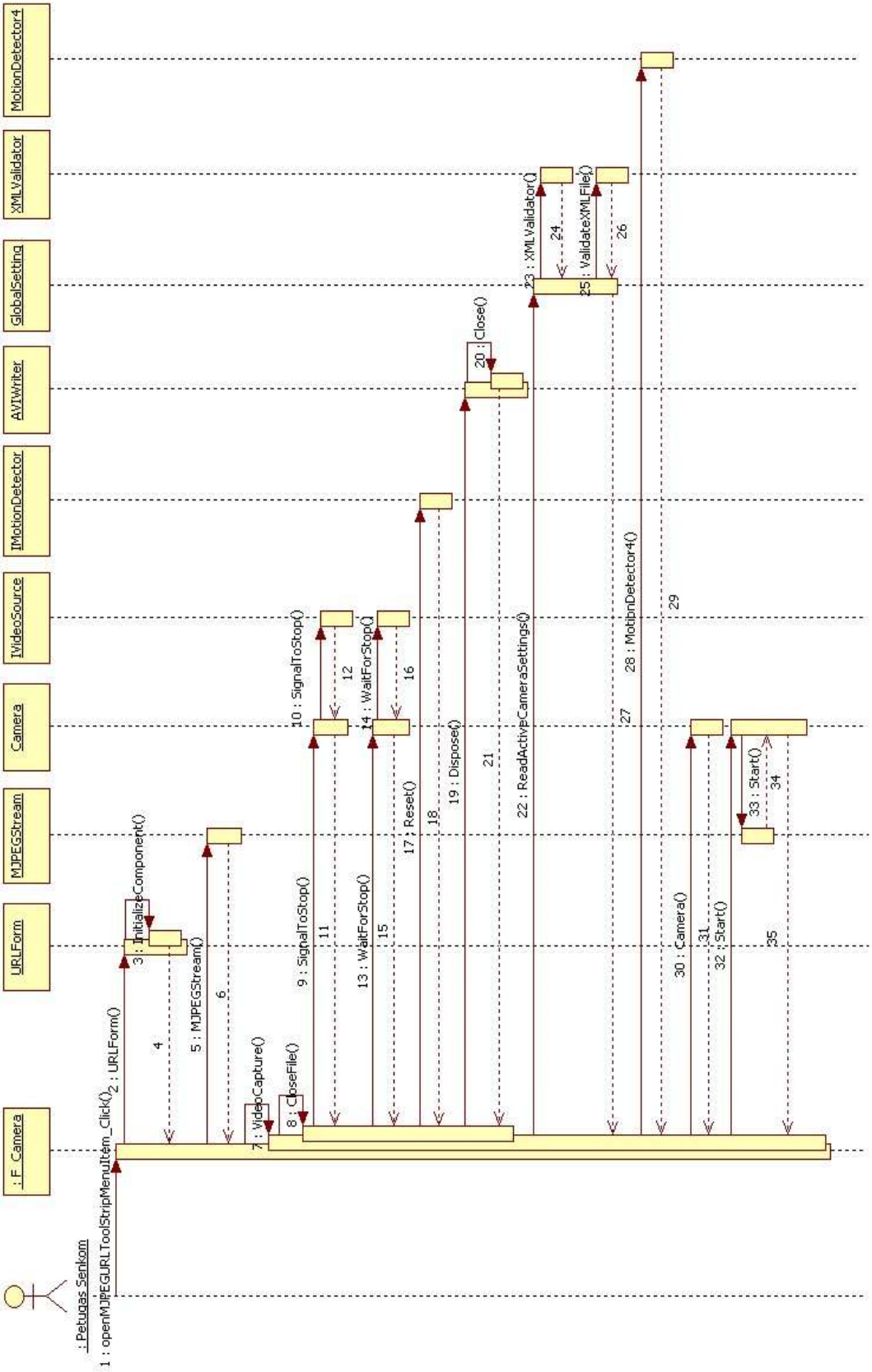
Gambar III.35 *Sequence Diagram Login*



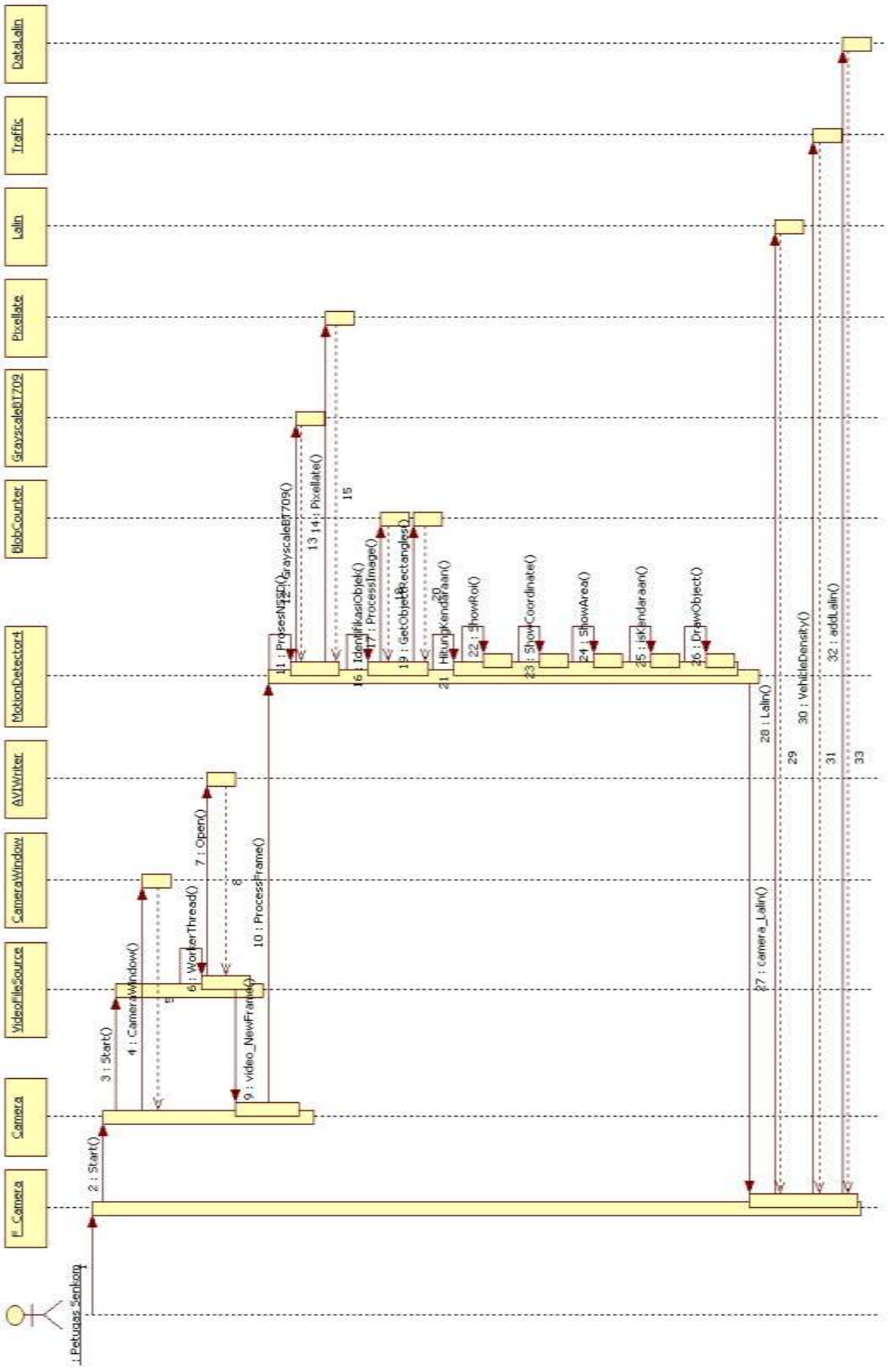
Gambar III.36 ***Sequence Diagram Setting***



Gambar III.37 ***Sequence Diagram Show Options***



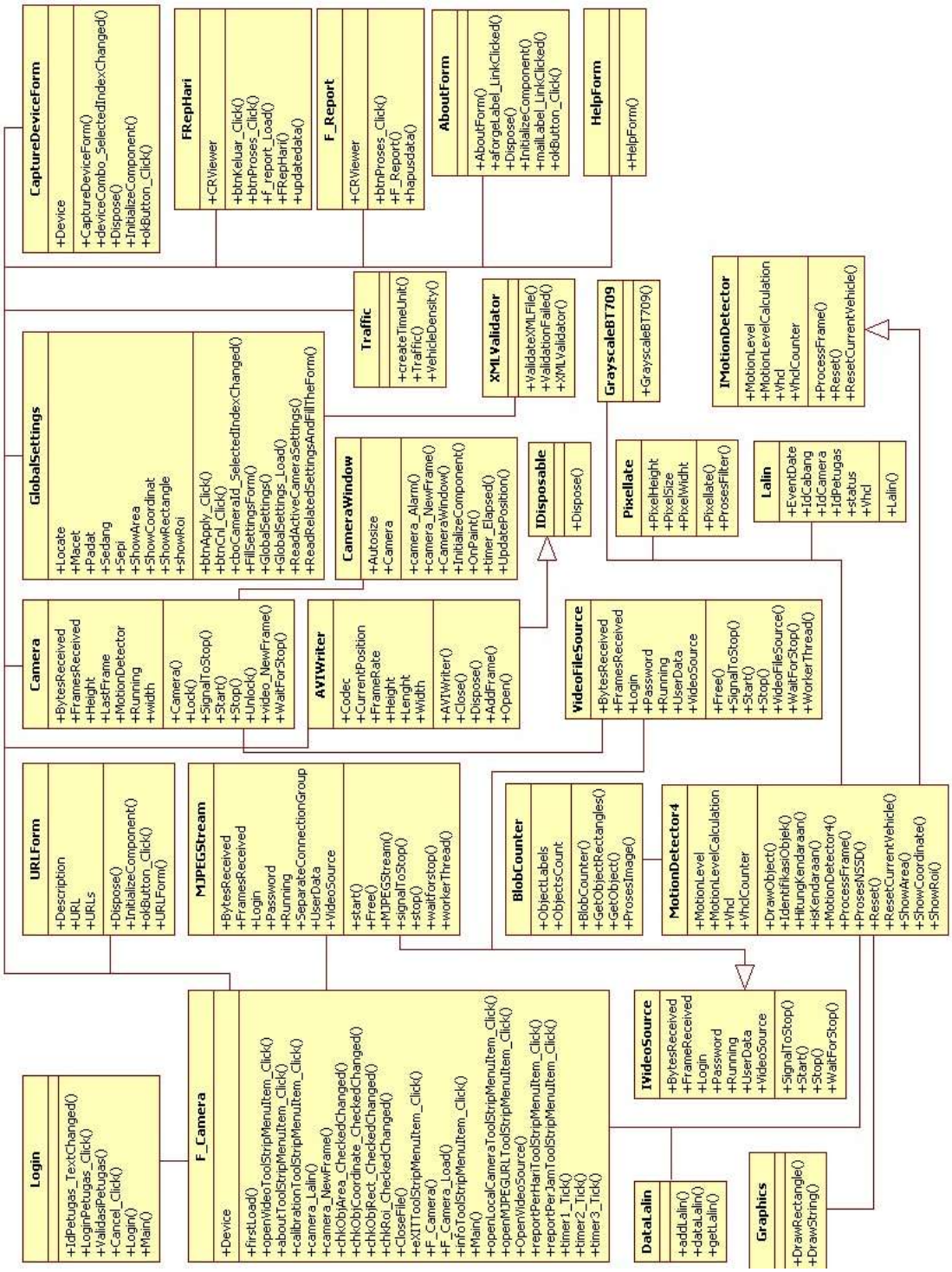
Gambar III.38 ***Sequence Diagram Streaming Video CCTV***



Gambar III.39 *Sequence Diagram* Pengolahan Citra dan Analisa Citra

3.4.4.4 *Class Diagram*

Class diagram menjelaskan analisis *Class* yang akan digunakan. *Class* diagram sistem ini digambarkan pada Gambar III.36:



Gambar III.40 *Class Diagram*

3.5 Perancangan Sistem

Perancangan sistem menjelaskan perancangan *Class*, *database* dan antarmuka yang akan dibangun pada sistem ini.

3.5.1 Perancangan *Class*

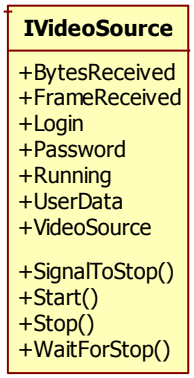
Perancangan *Class* menjelaskan *Class*-*Class* yang akan dibuat. Terdapat 3 jenis *Class* yaitu *Class Entity*, *Class* boundary dan *Class* control.

3.5.1.1 Perancangan *Class* Entity (Data)

Class entity merupakan *Class* yang berisi data atau segala sesuatu (*concrete*, *conceptual*, event, dan *state*) dapat dijadikan suatu entity dalam suatu *Class*. Terdapat 9 *Class* entity, yaitu:

1. **Perancangan *Class* IvideoSource**

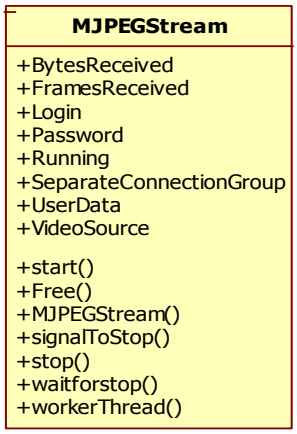
   Class IvideoSource merupakan *Class* entity yang berisi data sumber video, perancangan *Class* IvideoSource dapat dilihat pada gambar berikut:



Gambar III.41 Perancangan *Class* IvideoSource

2. **Perancangan *Class* MJPEGStream**

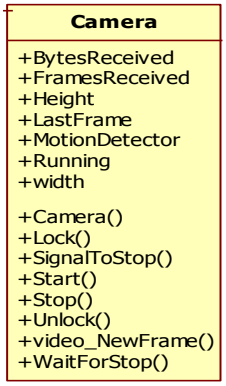
   Class MJPEGStream merupakan *Class* entity yang berisi data streaming video, perancangan *Class* MJPEGStream dapat dilihat pada gambar berikut:



Gambar III.42 Perancangan *Class* MJPEGStream

3. Perancangan *Class Camera*

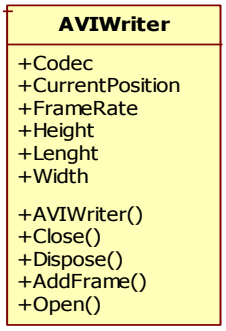
Class Camera merupakan *Class* entity yang berisi data frame citra, perancangan *Class Camera* dapat dilihat pada gambar berikut:



Gambar III.43 Perancangan *Class Camera*

4. Perancangan *Class* AVIWriter

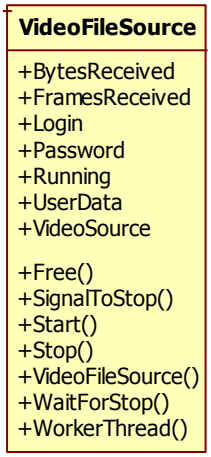
Class AVIWriter merupakan *Class* entity yang berisi data pemrosesan video, perancangan *Class* AVIWriter dapat dilihat pada gambar berikut:



Gambar III.44 Perancangan *Class* AVIWriter

5. Perancangan *Class* VideoFileSource

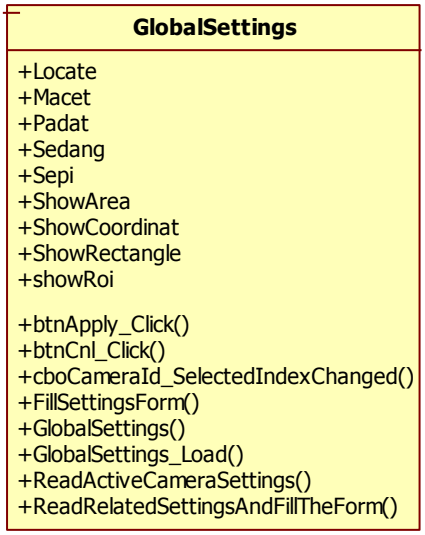
Class VideoFileSource merupakan *Class* entity yang berisi data sumber video, perancangan *Class* VideoFileSource dapat dilihat pada gambar berikut:



Gambar III.45 Perancangan *Class* VideoFileSource

6. Perancangan *Class* GlobalSettings

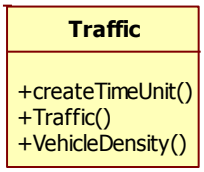
Class GlobalSettings merupakan *Class* entity yang berisi data-data dari *setting* ROI, *setting density* dan *setting camera*, perancangan *Class* GlobalSettings dapat dilihat pada gambar berikut:



Gambar III.46 Perancangan *Class* GlobalSettings

7. Perancangan *Class* Traffic

Class Traffic merupakan *Class* entity yang berisi data tingkat kepadatan, perancangan *Class* Traffic dapat dilihat pada gambar berikut:



Gambar III.47 Perancangan *Class* Traffic

8. Perancangan *Class* Lalin

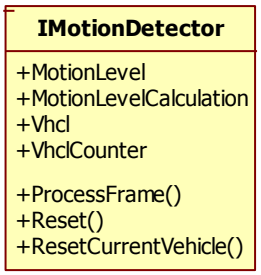
Class Lalin merupakan *Class* entity yang berisi data lalin, perancangan *Class* Lalin dapat dilihat pada gambar berikut:



Gambar III.48 Perancangan *Class* Lalin

9. Perancangan *Class* ImotionDetector

Class ImotionDetector merupakan *Class* entity yang berisi data hasil perhitungan objek dan *motion level*, perancangan *Class* ImotionDetector dapat dilihat pada gambar berikut:



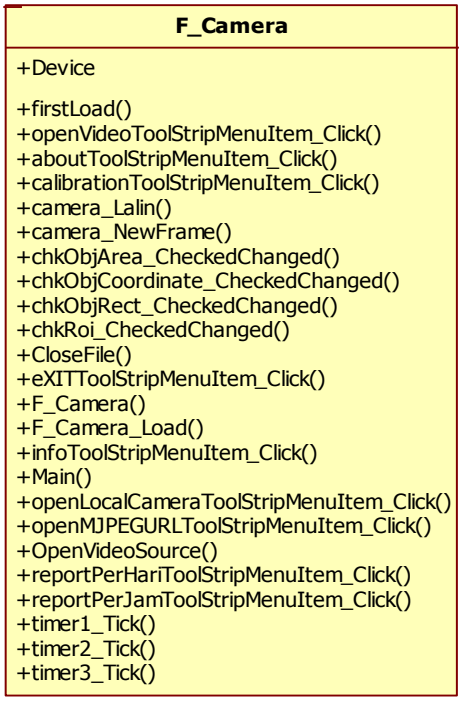
Gambar III.49 Perancangan *Class* ImotionDetector

3.5.1.2 Perancangan *Class* Boundary (Interface)

Class boundary merupakan *Class* yang berisi interface. Pada sistem ini Terdapat 7 *Class* boundary, yaitu:

1. **Perancangan *Class* F_Camera**

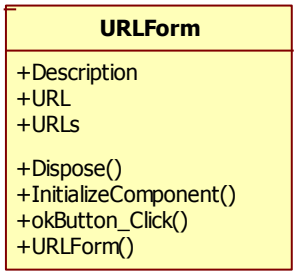
   Class F_Camera merupakan *Class* interface yang menampilkan semua proses dari yang dijalankan pada sistem ini, perancangan *Class* F_Camera dapat dilihat pada gambar berikut:



Gambar III.50 Perancangan *Class* F_Camera

2. **Perancangan *Class* URLForm**

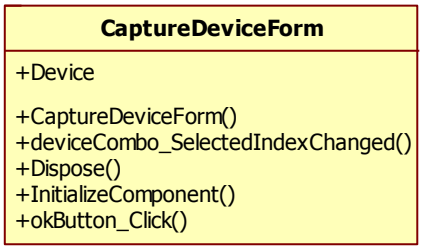
   Class URLForm merupakan *Class* interface yang menampilkan alamat URL, perancangan *Class* URLForm dapat dilihat pada gambar berikut:



Gambar III.51 Perancangan *Class* URLForm

3. Perancangan *Class* CaptureDeviceForm

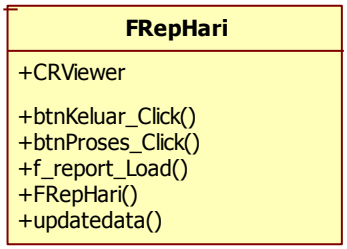
Class CaptureDeviceForm merupakan *Class* interface yang menampilkan form untuk pemilihan video, perancangan *Class* CaptureDeviceForm dapat dilihat pada gambar berikut:



Gambar III.52 Perancangan *Class* CaptureDeviceForm

4. Perancangan *Class* FrepHari

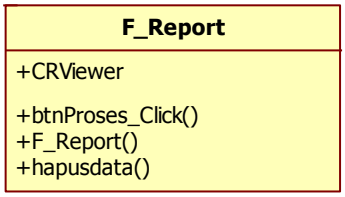
Class FrepHari merupakan *Class* interface yang menampilkan form report per hari, perancangan *Class* FrepHari dapat dilihat pada gambar berikut:



Gambar III.53 Perancangan *Class* FrepHari

5. Perancangan *Class* F_Report

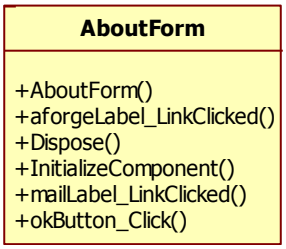
Class F_Report merupakan *Class* interface yang menampilkan form report per jam, perancangan *Class* F_Report dapat dilihat pada gambar berikut:



Gambar III.54 Perancangan *Class* F_Report

6. Perancangan *Class* AboutForm

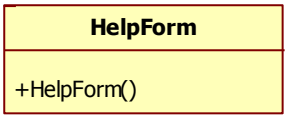
Class AboutForm merupakan *Class* interface yang menampilkan form pembuat aplikasi, perancangan *Class* AboutForm dapat dilihat pada gambar berikut:



Gambar III.55 Perancangan *Class* AboutForm

7. Perancangan *Class* HelpForm

Class HelpForm merupakan *Class* interface yang menampilkan form bantuan, perancangan *Class* HelpForm dapat dilihat pada gambar berikut:



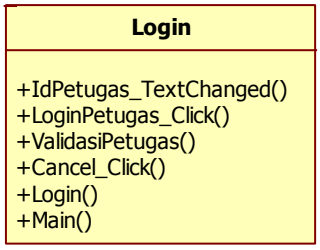
Gambar III.56 Perancangan *Class* HelpForm

3.5.1.3 Perancangan *Class* Control

Class Control merupakan suatu *Class* yang difungsikan untuk mengatur Entity *Class* dan Interfaaces *Class*. Pada sistem ini terdapat 10 buah *Class* control, yaitu:

1. Perancangan *Class* Login

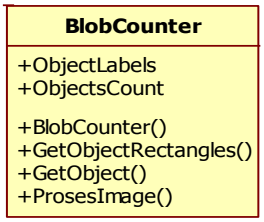
Perancangan *Class* Login dapat dilihat pada gambar berikut:



Gambar III.57 Perancangan *Class* Login

2. Perancangan *Class* BlobCounter

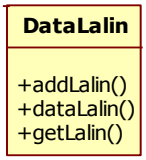
Perancangan *Class* BlobCounter dapat dilihat pada gambar berikut:



Gambar III.58 Perancangan *Class* BlobCounter

3. Perancangan *Class* DataLalin

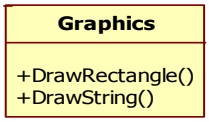
Perancangan *Class* DataLalin dapat dilihat pada gambar berikut:



Gambar III.59 Perancangan *Class* DataLalin

4. Perancangan *Class* Graphics

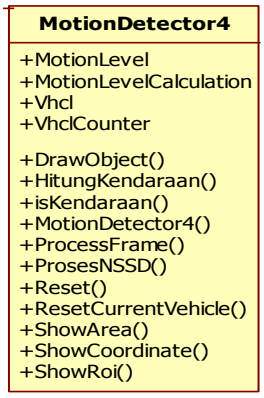
Perancangan *Class* Graphics dapat dilihat pada gambar berikut:



Gambar III.60 Perancangan *Class* Graphics

5. Perancangan *Class* MotionDetector4

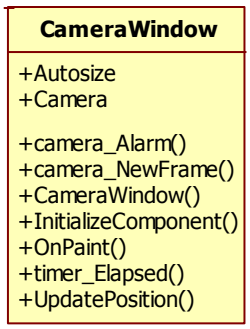
Perancangan *Class* MotionDetector4 dapat dilihat pada gambar berikut:



Gambar III.61 Perancangan *Class* MotionDetector4

6. Perancangan *Class* CameraWindow

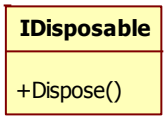
Perancangan *Class* CameraWindow dapat dilihat pada gambar berikut:



Gambar III.62 Perancangan *Class* CameraWindow

7. Perancangan *Class* Idisposable

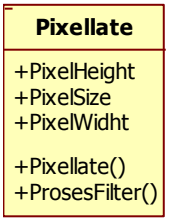
Perancangan *Class* Idisposable dapat dilihat pada gambar berikut:



Gambar III.63 Perancangan *Class* Idisposable

8. Perancangan *Class* Pixellate

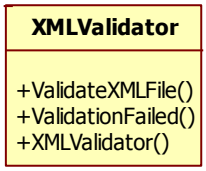
Perancangan *Class* Pixellate dapat dilihat pada gambar berikut:



Gambar III.64 Perancangan *Class* Pixellate

9. Perancangan *Class* XMLValidator

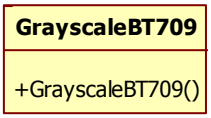
Perancangan *Class* XMLValidator dapat dilihat pada gambar berikut:



Gambar III.65 Perancangan *Class* XMLValidator

10. Perancangan *Class* GrayscaleBT709

Perancangan *Class* GrayscaleBT709 dapat dilihat pada gambar berikut:



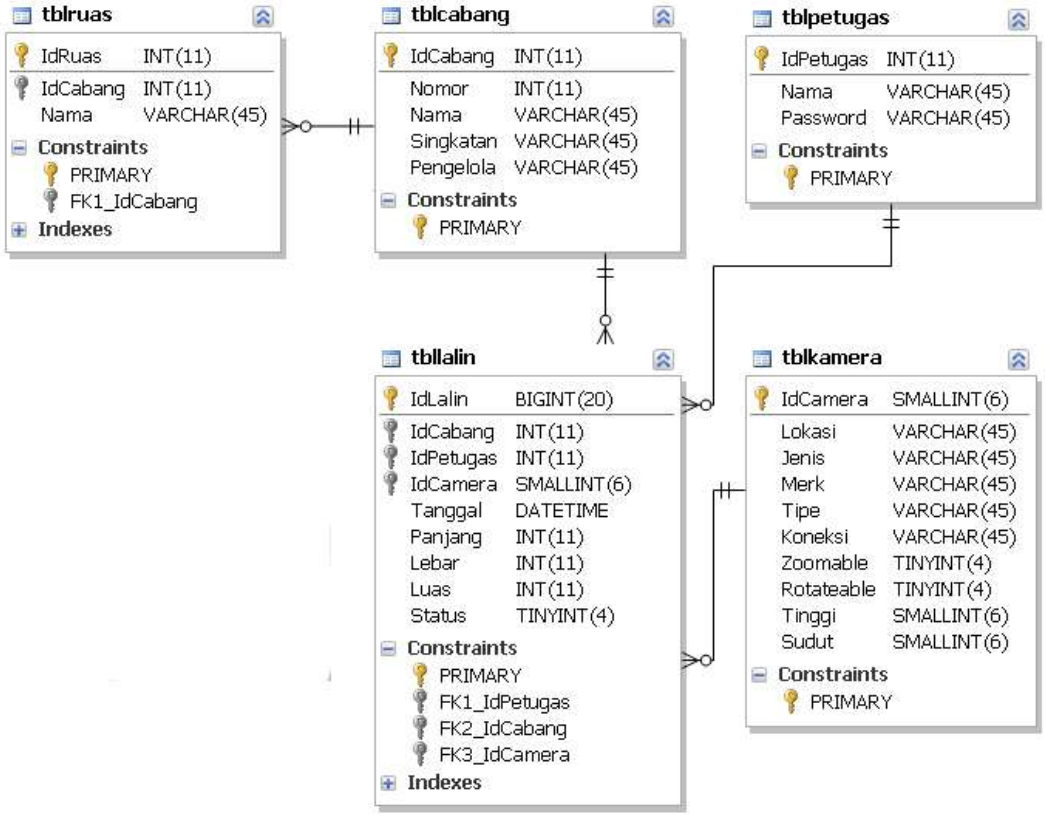
Gambar III.66 Perancangan *Class* GrayscaleBT709

3.5.2 Perancangan *Database*

Perancangan *database* menjelaskan basis data yang akan dibuat pada sistem ini.

3.5.2.1 Diagram Relasi

Diagram relasi menggambarkan relasi setiap tabel di dalam *database*, seperti terlihat pada gambar berikut:



Gambar III.67 Diagram Relasi *Database*

3.5.2.2 Struktur Tabel *Database*

Struktur tabel *database* menjelaskan deskripsi setiap tabel yang digunakan pada sistem ini.

1. Tabel Ruas

   Tabel ruas menyimpan data tentang nama ruas jalan tol, struktur data tabel ruas dapat dilihat pada tabel berikut ini:

Tabel III.12 Struktur Tabel Ruas

Field	Type	Keterangan
IdRuas	INT (11)	PRIMARY KEY
IdCabang	INT (11)	FOREIGN KEY, Ref tblcabang(idCabang)
Nama	VARCHAR (45)	

2. Tabel Cabang

   Tabel cabang menyimpan data tentang nama cabang jalan tol, struktur data tabel cabang dapat dilihat pada tabel berikut ini:

Tabel III.13 Struktur Tabel Cabang

Field	Type	Keterangan
IdCabang	INT (11)	PRIMARY KEY
Nama	VARCHAR (45)	
Singkatan	VARCHAR (45)	
Pengelola	VARCHAR (45)	

3. Tabel Lalin

   Tabel lalin menyimpan data lalu lintas kendaraan , struktur data tabel lalin dapat dilihat pada tabel berikut ini:

Tabel III.14 Struktur Tabel Lalin

Field	Type	Keterangan
IdLalin	BIGINT (20)	PRIMARY KEY
Tanggal	DATETIME	
IdCabang	INT (11)	FOREIGN KEY, Ref tblcabang(idCabang)
IdCamera	SMALLINT (6)	FOREIGN KEY, Ref tblkamera(IdCamera)
IdPetugas	INT (11)	FOREIGN KEY, Ref tblpetugas(IdPetugas)
Panjang	INT (11)	
Lebar	INT (11)	
Luas	INT (11)	

Status	TINYINT (4)	

4. Tabel Petugas

Tabel petugas menyimpan data petugas yang bertugas dalam pemantauan lalu lintas kendaraan, struktur data tabel petugas dapat dilihat pada tabel berikut ini:

Tabel III.15 Struktur Tabel Petugas

Field	Type	Keterangan
IdPetugas	INT (11)	PRIMARY KEY
Nama	VARCHAR (45)	
Password	VARCHAR (45)	

5. Tabel Kamera

Tabel kamera menyimpan data-data kamera CCTV jalur yang terpasang, struktur data tabel kamera dapat dilihat pada tabel berikut ini:

Tabel III.16 Struktur Tabel Kamera

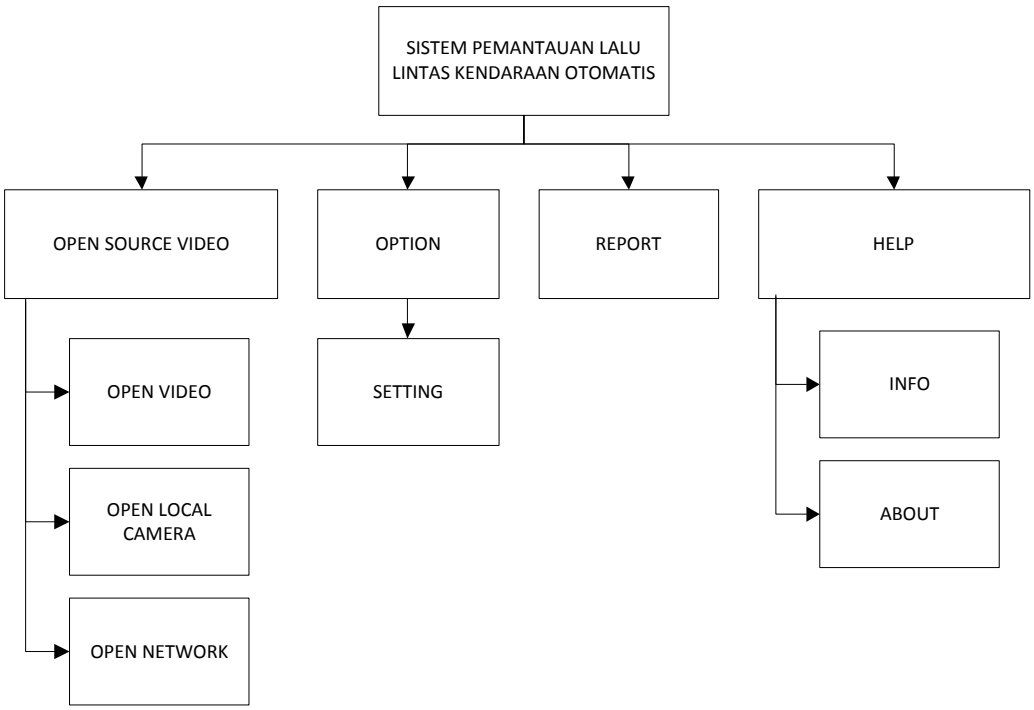
Field	Type	Keterangan
IdCamera	SMALLINT (6)	PRIMARY KEY
Lokasi	VARCHAR (45)	
Jenis	VARCHAR (45)	
Merk	VARCHAR (45)	
Tipe	VARCHAR (45)	
Koneksi	VARCHAR (45)	
Zoomable	TINYINT (4)	
Rotateable	TINYINT (4)	
Tinggi	SMALLINT (6)	
Sudut	SMALLINT (6)	

3.5.3 Perancangan Antarmuka

Perancangan Antarmuka menjelaskan antarmuka yang akan dibuat pada sistem ini. Perancangan antarmuka terdiri dari 2 sub judul yaitu perancangan menu dan perancangan form.

3.5.3.1 Perancangan Menu

Perancangan menu menjelaskan struktur menu yang akan dibuat pada suatu sistem. Gambar berikut menunjukkan perancangan menu pada sistem ini.



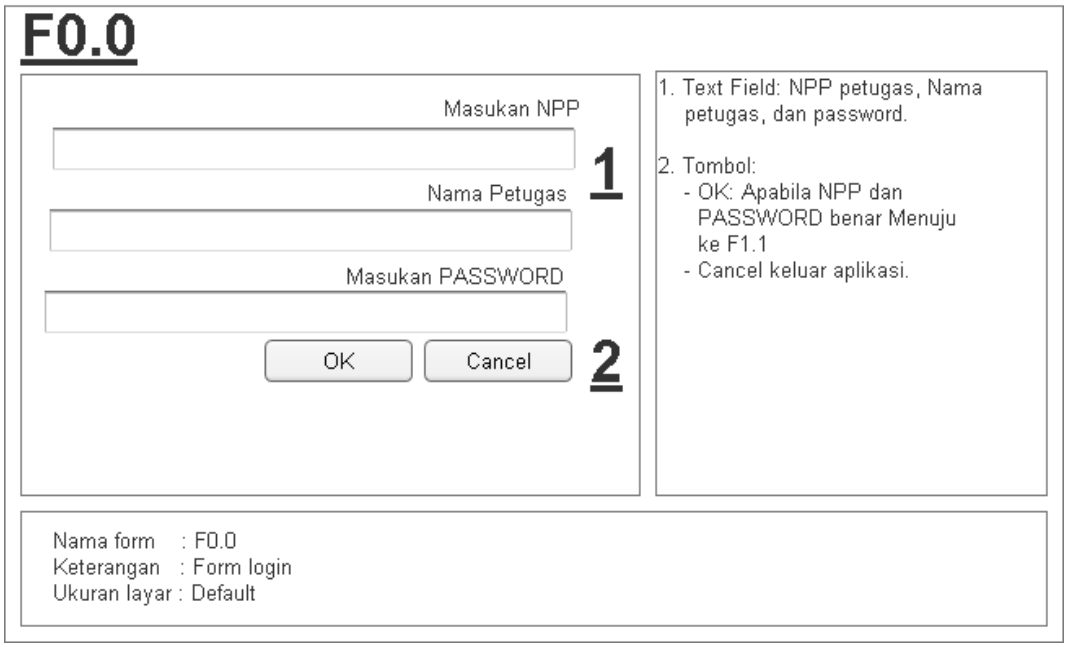
Gambar III.68 Struktur Menu Aplikasi Pemantauan Lalu Lintas Kendaraan Otomatis

3.5.3.2 Perancangan Form

Perancangan form dibagi menjadi 9 rancangan antarmuka. Rancangan form tersebut yaitu: Perancangan Form Login(F0.0), Perancangan Form Tampilan Utama(F1.1), Perancangan Form *Open Video*(F1.2), Perancangan Form *Open Local Camera*(F1.3), Perancangan Form *Open Network Camera*(F1.3), Perancangan Form *Setting*(F2.1), Perancangan Form *Report*(F3.1), Perancangan Form *help*(F4.1), Perancangan Form *About*(F4.1)

A. Perancangan Form Login

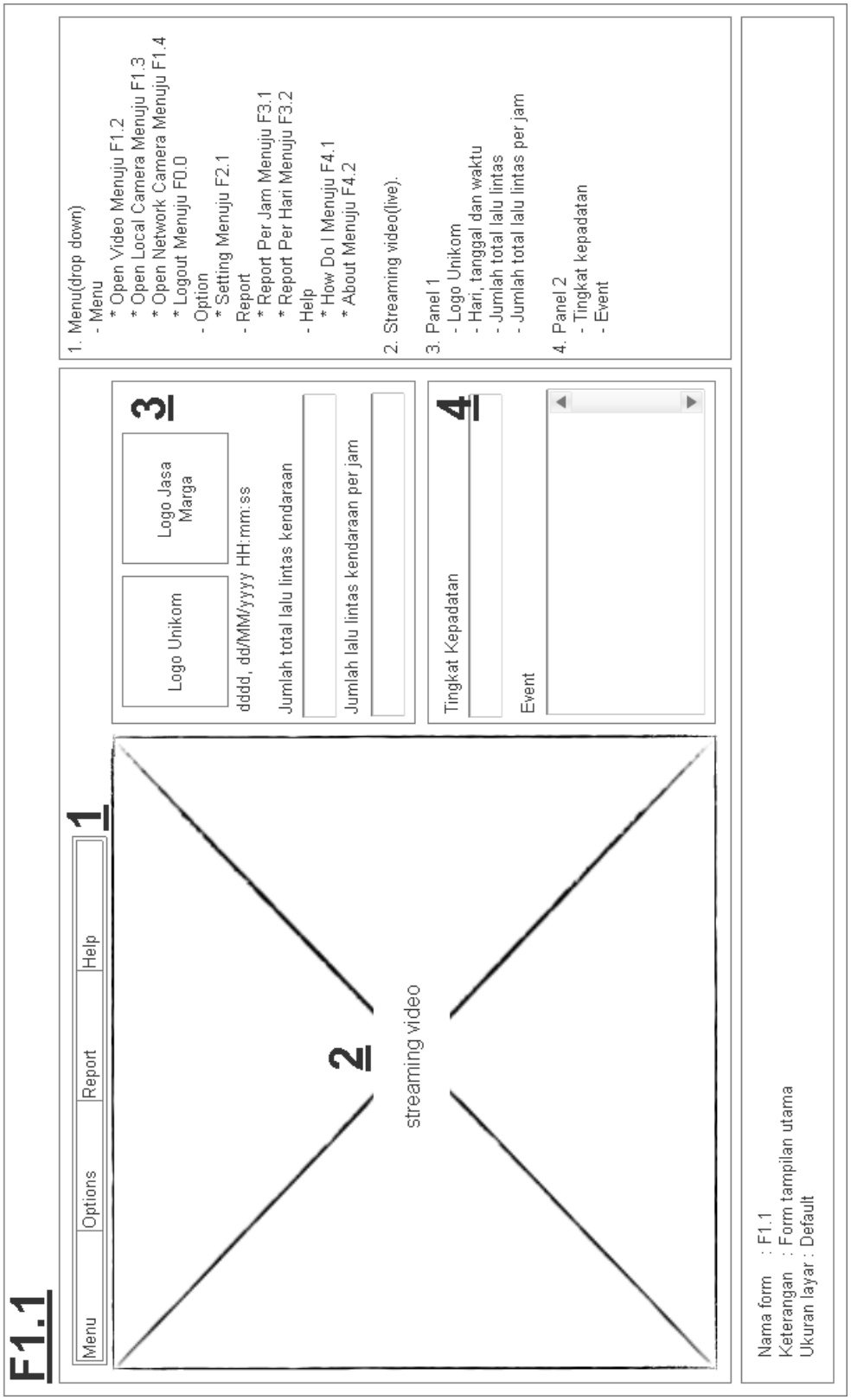
Perancangan form login menampilkan tampilan awal program aplikasi. Tampilan ini untuk menginput NPP Petugas dan password. Perancangan form login dapat dilihat pada Gambar berikut:



Gambar III.69 Form Login(F0.0)

B. Perancangan Form Tampilan Utama

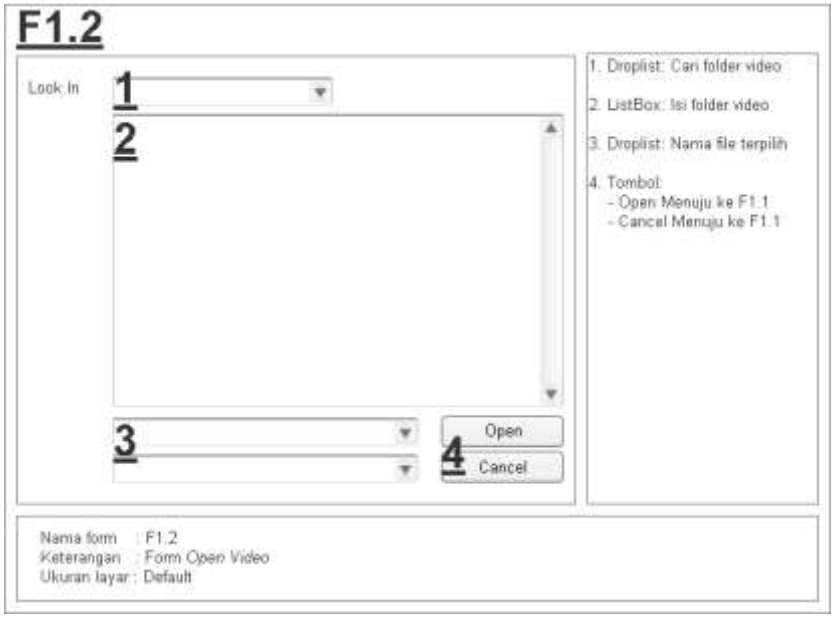
Perancangan form tampilan utama menampilkan menu-menu yang dapat diakses oleh user, tampilan video dan informasi hasil dari pengolahan citra/nssd berupa tampilan informasi kepadatan. Perancangan form tampilan utama dapat dilihat pada berikut:



Gambar III.70 Form Tampilan Utama(F1.1)

C. Perancangan Form *Open Video*

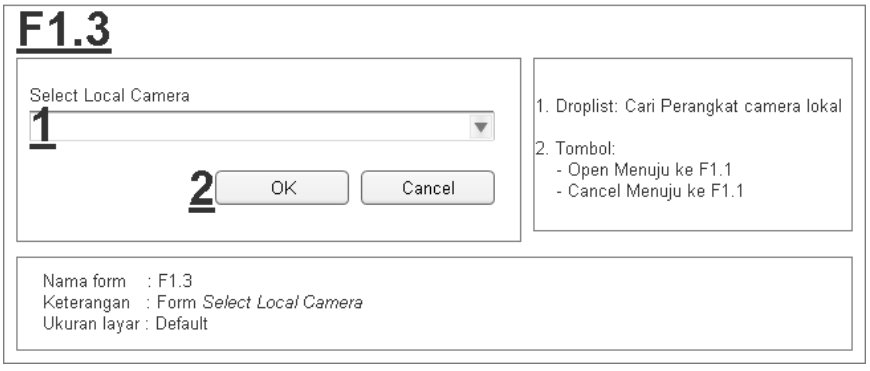
Perancangan form *Open Video* menampilkan tampilan untuk membuka file video yang tersimpan di media penyimpanan. perancangan form *Open Video* dapat dilihat pada berikut:



Gambar III.71 Form *Open Video*(F1.2)

D. Perancangan Form *Open Local Camera*

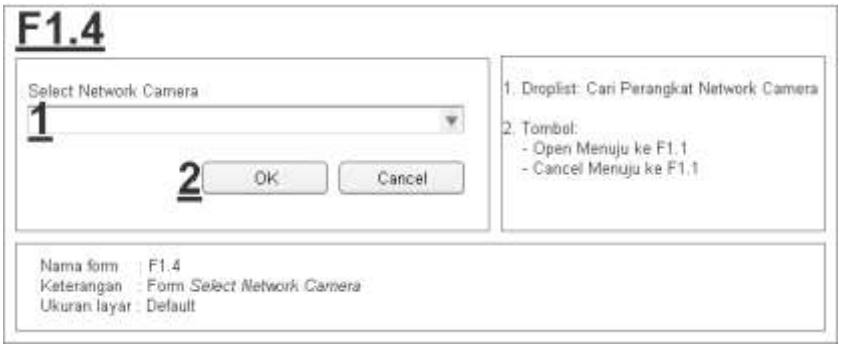
Perancangan form *Local Camera* menampilkan tampilan untuk menampilkan *streaming video* pada kamera lokal yang terpasang langsung di komputer. perancangan form *Local Camera* dapat dilihat pada gambat berikut:



Gambar III.72 *Form Local Camera*(F1.3)

E. Perancangan Form *Open Network Camera*

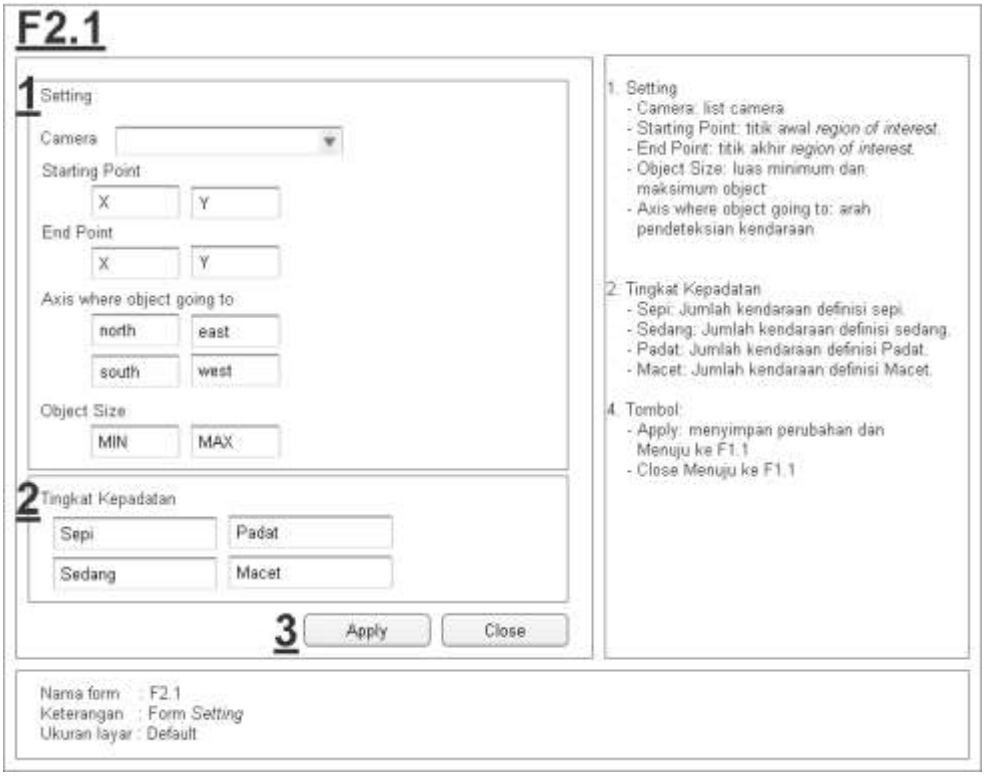
Perancangan form *Network Camera* menampilkan tampilan untuk menampilkan streaming video pada IP Camera yang terpasang di jaringan komputer. perancangan form *Network Camera* dapat dilihat pada gambat berikut:



Gambar III.73 Form *Network Camera* (F1.4)

F. Perancangan Form *Setting*

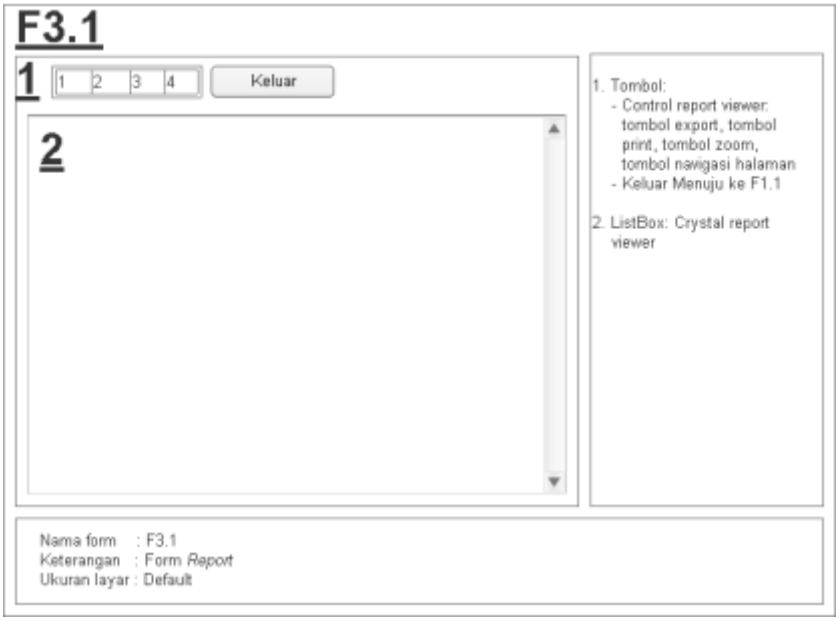
Perancangan form *Setting* menampilkan tampilan untuk pengaturan nilai dari koordinat *region of interest*(ROI), pengaturan nilai arah bergerak dari objek dan pengaturan nilai kepadatan lalu lintas per jam. perancangan form *Setting* dapat dilihat pada gambat berikut:



Gambar III.74 Form *Setting*(F2.1)

G. Perancangan Form *Report*

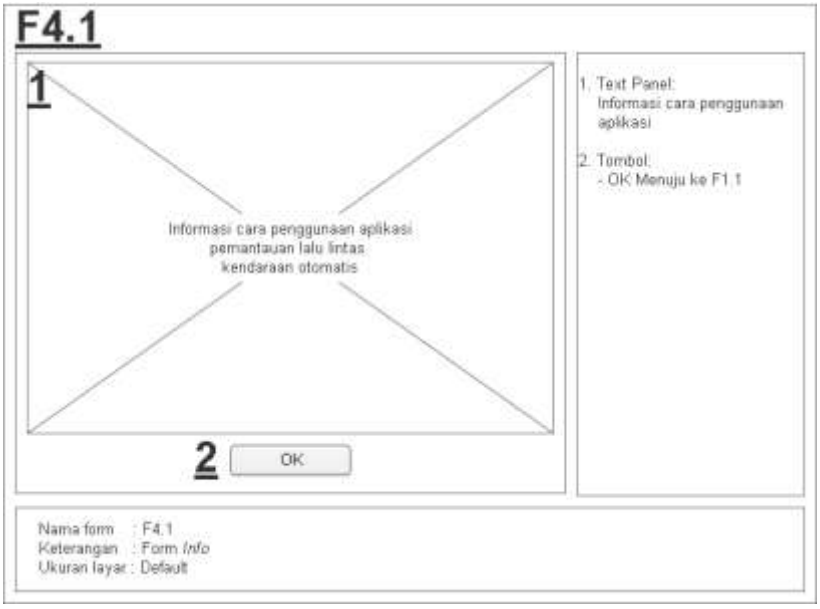
Perancangan form *Report* menampilkan tampilan reporting kepadatan lalu lintas kendaraan. Perancangan form *Report* dapat dilihat pada gambat berikut:



Gambar III.75 Form *Report*(F3.1)

H. Perancangan Form *Help*

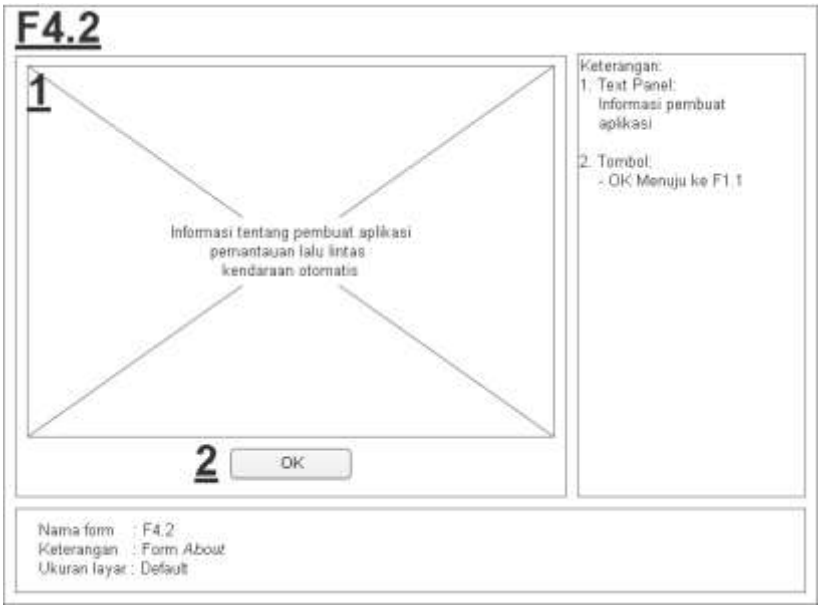
Perancangan form *Help* menampilkan tampilan informasi cara penggunaan aplikasi pemantauan kepadatan lalu lintas. Perancangan form *Help* dapat dilihat pada gambat berikut:



Gambar III.76 Form *help*(F4.1)

I. Perancangan Form *About*

Perancangan form about menampilkan tampilan informasi tentang pembuat aplikasi pemantauan kepadatan lalu lintas. Perancangan form about dapat dilihat pada gambat berikut:



Gambar III.77 Form *About*(F4.2)