BAB 2

LANDASAN TEORI

2.1 Sistem Pendukung Keputusan (SPK)

2.1.1 Pengertian SPK

SPK pertama kali diungkapkan pada awal tahun 1970-an oleh Scott Morton, yaitu sebagai suatu sistem yang berbasis komputer yang ditujukan untuk membantu pengambil keputusan dalam memanfaatkan data dan model tertentu untuk memecahkan berbagai persoalan yang tidak terstruktur (Turban, 2005 : 19). Setelah itu pengertian SPK mulai berkembang, berikut ini beberapa pengertian dari SPK :

a. Sekumpulan model dasar dari prosedur dalam pemrosesan data dan kebijakan untuk membantu seorang manajer dala pembuatan keputusan (Turban dan Aronson, 2001).
b. Sistem yang memiliki lima karakteristik utama (Sparague Carlson, 1993) :
    1) sistem yang berbasis komputer;
    2) dipergunakan untuk membantu para pengambil keputusan;
    3) untuk memecahkan masalah-masalah rumit yang mustahil dilakukan dengan kalkulasi manual;
    4) melalui cara simulasi yang interaktif; dan
    5) data dan model analisis sebagai komponen utama.

Dari beberapa definisi di atas dapat disimpulkan bahwa sistem pendukung keputusan adalah suatu sistem berbasis komputer yang mengkombinasikan data dan model dengan tujuan membantu para pengambil keputusan untuk menyelesaikan masalah-masalah yang semi terstruktur maupun yang tidak terstruktur melalui cara simulasi interaktif.

Pengambilan keputusan dilakukan dengan pendekatan sistematis terhadap permasalahan melalui proses pengumpulan data yang diolah menjadi informasi serta ditambah dengan faktor-faktor yang perlu dipertimbangkan dalam

pengambilan keputusan. Dalam penerapannya SPK akan memberikan beberapa solusi penyelesaian untuk masalah yang sedang dihadapi, namun setiap keputusan yang nantinya akan diambil merupakan kebijakan para pengambil keputusan. SPK tidak memaksa para pengambil keputusan untuk mengambil keputusan berdasarkan salah satu dari solusi yang ditawarkan. Dengan kata lain SPK meningkatkan efektifitas bukan efisiensi pengambilan keputusan.

2.1.2 Tujuan SPK

Tujuan pembentukan SPK yang efektif adalah memanfaatkan keunggulan dua unsur, yaitu manusia dan perangkat elektronik. Jika terlalu banyak menggunakan komputer akan menghasilkan pemecahan yang dangkal, tetapi jika terlalu banyak mengandalkan manusia akan memunculkan reaksi yang lamban, pemanfaatan data yang serba terbatas, dan kelambanan dalam mengkaji alternatif yang relevan (Kadarsah Suryadi, 1998 : 2).

2.1.3 Tahap-tahap Pengambilan Keputusan

Menurut Simon (1960), tahap-tahap yang harus dilalui dalam proses pengambilan keputusan adalah sebagai berikut (Kardasah Suryadi, 1998:15 – 16) :

a. *Intelligence*

   Tahap ini merupakan proses penelusuran dan pendeteksian dari lingkup problematika serta proses pengenalan masalah. Data masukan diperoleh, diproses, dan diuji daam rangka mengidentifikasikan masalah.

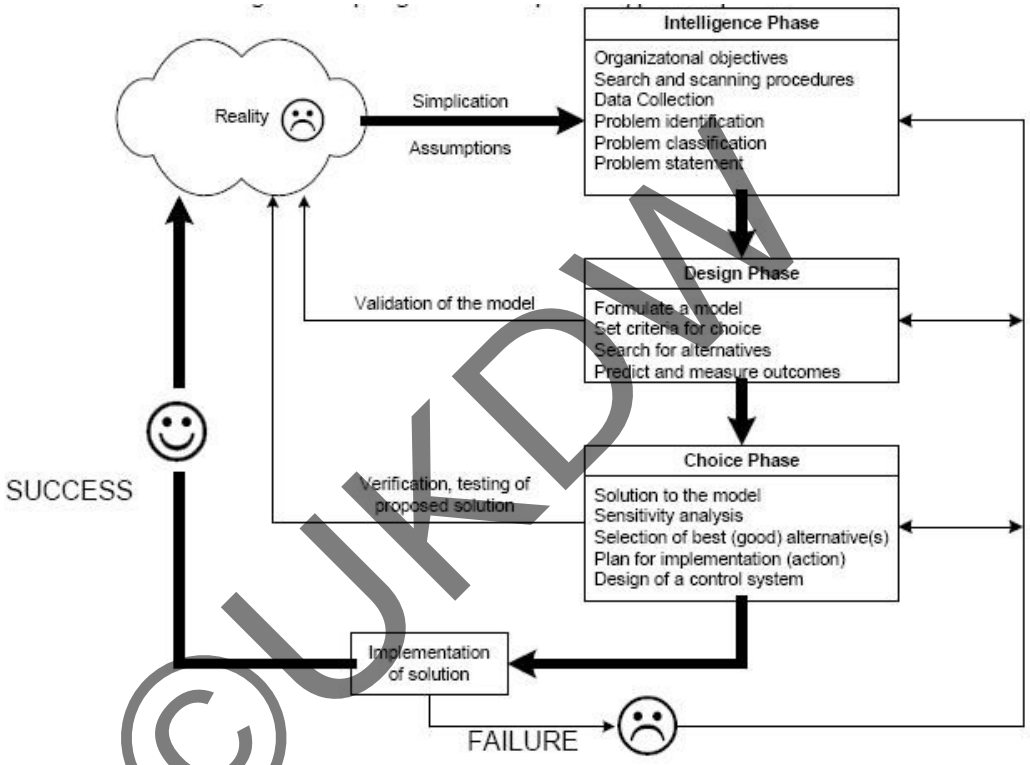
b. *Design*

   Tahap ini merupakan proses menemukan, mengembangkan dan menganalisa alternatif tindakan yang bisa dilakukan. Tahap ini meliputi proses untuk mengerti masalah, menurunkan solusi dan menguji kelayakan solusi.

c. *Choice*

Pada tahap ini dilakukan proses pemilihan dengan berbagai alternatif tindakan yang mungkin dijalankan. Hasil pemilihan tersebut kemudian diimplementasikan dalam proses pengambilan keputusan.

Tabel 2.1 di bawah ini mengilustrasikan tahap-tahap yang dilakukan dalam pengambilan keputusan.



Gambar 2.1. Tahap-tahap Pengambilan Keputusan
Dikutip dari : Turban, E., Aronson, J.E., Liang, T.P., (2005, 41).

2.1.4 Komponen – komponen SPK

Dalam buku Turban, E., Aronson, J.E., dan Liang , T.P. (2005), disebutkan bahwa SPK terdiri dari beberapa subsistem :

a. Subsistem Manajemen Data

Subsistem manajemen data memasukkan satu database yang relevan untuk situasi dan dikelola oleh piranti lunak disebut sistem manajemen *database*

(DBMS). Subsistem manajemen data dapat diinterkoneksikan dengan data *warehouse* perusahaan, suatu *repositori* untuk data perusahaan yang relevan untuk pengambilan keputusan. Biasanya data disimpan atau diakses via *server database.*

b. Subsistem Manajemen Model

Subsistem manajemen model merupakan paket piranti lunak yang memasukkan model keuangan, statistik, ilmu manajemen, atau model kuantitatif lainnya. Semua itu memberikan kapabilitas analitik dan manajemen piranti lunak yang tepat. Bahasa pemodelan yang membangun model kostum juga dimasukan. Piranti lunak ini sering disebut manajemen basis model (MBMS). Komponen ini dapat dikoneksikan ke penyimpanan koorporat atau eksternal yang ada pada model.

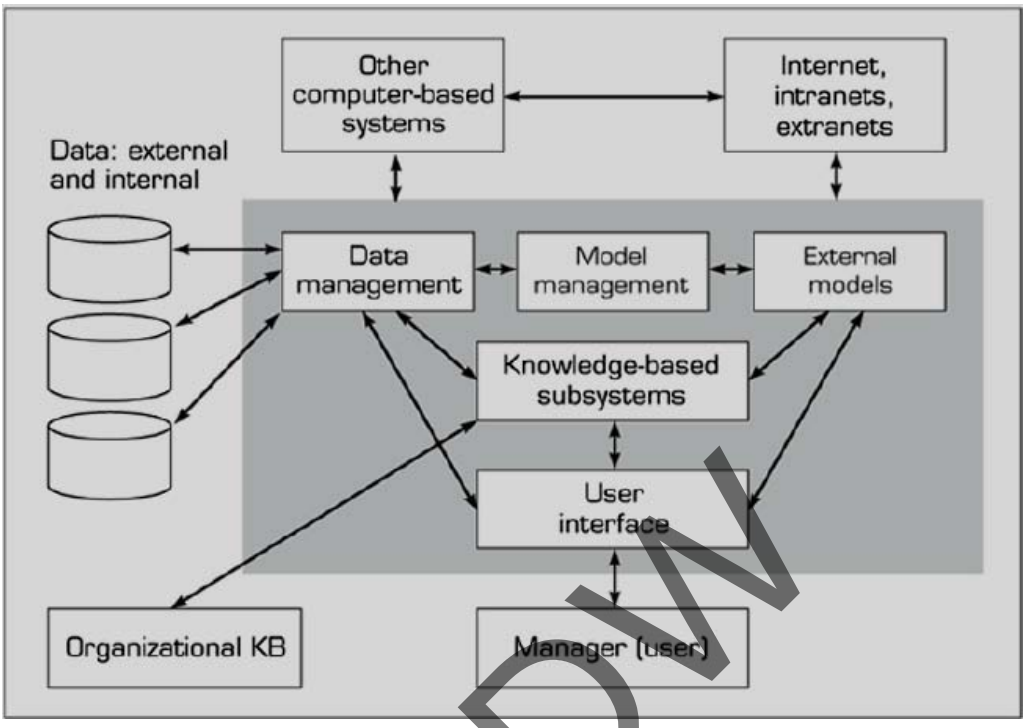
c. Subsistem Antarmuka Pengguna

Pengguna berkomunikasi dengan memerintahkan SPK melalui subsistem ini. Pengguna adalah bagian yang dipertimbangkan dari sistem. Para peneliti menegaskan bahwa beberapa kontribusi unik dari SPK berasal dari interaksi yang intensif antara komputer dengan pembuat keputusan.

d. Subsistem Manajemen Berbasis Pengetahuan

Subsistem ini dapat mendukung semua subsistem lain atau bertindak sebagai suatu komponen independen. Subsistem ini dapat diinterkoneksikan dengan repositori pengetahuaan perusahaan atau bagian dari sistem manajemen pengetahuan yang kadang-kadang disebut basis pengetahuan organisasional.

Berdasarkan definisi, SPK harus mencakup tiga komponen utama dari DBMS, MBMS, dan antarmuka pengguna. Subsistem manajemen berbasis pengetahuan adalah opsional, namun dapat memberi banyak manfaat karena intelegensi bagi ketiga komponen utama. Komponen-komponen SPK terlihat pada Gambar 2.2.



Gambar 2.2. Komponen-komponen SPK
Dikutip dari : Turban, E., Aronson, J.E., Liang, T.P., (2005, 143).

2.2 Multi Criteria Decision Making (MCDM)

2.2.1 Pengertian MCDM

Kusumadewi (2006) dalam bukunya manyatakan bahwa Multi-Criteria Decision Making (MCDM) adalah suatu metode pengambilan keputusan untuk menetapkan alternatif terbaik dari sejumlah alternatif berdasarkan beberapa kriteria tertentu. Kriteria biasanya berupa ukuran-ukuran atau aturan-aturan atau standar yang digunakan dalam pengambilan keputusan. Secara umum dapat dikatakan bahwa MCDM menyeleksi alternatif terbaik dari sejumlah alternatif. Sedangkan menurut Artana (2005), MCDM merupakan suatu metode pengambilan keputusan yang didasarkan atas teori-teori, proses-proses, dan metode analitik yang melibatkan ketidakpastian, dinamika, dan aspek kriteria jamak. Dalam metode optimasi konvensional, cakupan umumnya hanya dibatasi pada suatu kriteria pemilihan. Pilihan yang diambil adalah pilihan yang paling memenuhi fungsi obyektif. Namun, masalah yang dihadapi khususnya yang lebih

bersifat praktis tidaklah sesederhana itu. Ada kalanya pertimbangan-pertimbangan subjektif harus dimasukkan ke dalam proses pembuatan keputusan.

Menurut Janko (2005), terdapat beberapa fitur umum yang digunakan dalam MCDM, yaitu: (Kusumadewi, 2006)

a. Alternatif, alternatif adalah obyek-obyek yang berbeda dan memiliki kesempatan yang sama untuk dipilih oleh pengambil keputusan.
b. Atribut, atribut sering juga di sebut sebagai kriteria keputusan.
c. Bobot keputusan, bobot keputusan menunjukan kepentingan relatif dari setiap kriteria. $W = (w_1, w_2, w_3, ..., w_n)$.
d. Matriks keputusan, suatu matriks keputusan X yang berukuran *m x n*, berisi elemen-elemen x_{ij} yang merepresentasikan rating dari alternatif *Ai ; i* = 1, 2, 3, ..., *m* terhadap kriteria *Cj ; j* = 1, 2, 3, ..., *n*.

2.2.2 Sifat – Sifat dalam Memilih Kriteria

Sifat-sifat yang harus diperhatikan dalam memilih kriteria pada setiap persoalan pengambilan keputusan adalah sebagai berikut (Kardasah 1998:126).

a. Lengkap, sehingga dapat mencakup seluruh aspek penting dalam persoalan tersebut. Suatu set kriteria disebut lengkap apabila set ini dapat menunjukan seberapa jauh seluruh tujuan dapat dicapai.
b. Operasional, sehingga dapat digunakan dalam analisis. Sifat operasional mencakup beberapa pengertian, antara lain adalah bahwa kupulan kriteria harus mempunyai arti bagi pengambil keputusan, sehingga ia dapat benar-benar menghayati implikasinya terhadap alternatif yang ada. Selain itu, jika tujuan pengambilan keputusan ini harus dapat digunakan sebagai sarana untuk meyakinkan pihak lain, maka kumpulan kriteria ini harus dapat digunakan sebagai sarana untuk memberikan penjelasan atau untuk berkomunikasi. Operasional ini juga mencakup sifat dapat diukur. Pada dasarnya sifat dapat diukur ini adalah untuk memperoleh distribusi kemungkinan dari tingkat pencapaian kriteria yang mungkin diperoleh atau

dengan kata lain untuk keputusan dalam ketidakpastian, dan mengungkapkan preferensi pengambil keputusan atas pencapaitan kriteria.

c. Tidak berlebihan, sehingga menghindarkan perhitungan berulang. Dalam menentukan set kriteria, jangan sampai terdapat kriteria yang pada dasarnya mengandung pengertian yang sama.

d. Minimum, agar lebih mengkomperehensifkan persoalan. Dalam menentukan sejumlah kriteria perlu sedapat mungkin mengusahakan agar jumlah kriterianya sesedikit mungkin. Karena semakin banyak kriteria maka semakin sukar pula dalam hal menghayati persoalan dengan baik, dan jumlah perhitungan yang diperlukan dalam analisis akan meningkat dengan cepat.

MCDM menjadi rumit karena banyaknya kriteria yang terlibat dalam permasalahan. Pada permasalahan yang hanya melibatkan satu kriteria penilaian, proses pemilihan alternatif akan relatif lebih mudah walaupun terdapat banyak alternatif yang harus dipertimbangkan. Dengan demikian bisa dikatakan bahwa tingkat kesulitan pengambilan keputusan sensitif terhadap jumlah kriteria yang dipertimbangkan.

Tabucanon (1988) menyatakan bahwa suatu permasalahan tergolong MCDM jika dan hanya jika setidaknya terdapat dua kriteria yang saling bertentangan dan melibatkan dua solusi alternatif. Kriteria yang saling bertentangan (*conflicting criteria*) berarti kepuasan memilih suatu alternatif berdasarkan suatu kriteria tertentu akan berbeda berdasarkan kriteria yang lain. Sedangkan *non conflicting criteria* memperlihatkan adanya dominasi yang kuat dari suatu alternatif lain yang dibandingkan.

2.2.3 Kategori MCDM

a. *Multiple Objective Decision Making* (MODM)

MODM menyangkut masalah perancangan (*design*). Pada perancangan, teknik-teknik matematik optimasi digunakan untuk jumlah alternatif yang

sangat besar sampai dengan tak terhingga dan untuk menjawab pertanyaan apa (*what*) dan berapa banyak (*how much*).

b. *Mulitiple Attribute Decision Making* (MADM)

MADM menyangkut masalah pemilihan. Pada pemilihan, analisa matematis tidak terlalu banyak dibutuhkan atau dapat digunakan untuk pemilihan yang hanya memiliki alternatif dalam jumlah kecil.Metode *Analytical Hierarchy Process* (AHP) merupakan bagian dari teknik MADM.

2.3 The Weighted Sum Model (WSM)

The Weighted Sum Model (WSM) adalah pendekatan yang umum dipakai dalam mengambil keputusan alternatif terbaik, terutama pada permasalahan untuk dimensi tunggal (Evangelos Triantaphyllou, 2002).

Jika terdapat *M* alternatif dan *N* kriteria, maka untuk menentukan alternatif yang terbaik adalah dengan menggunakan persamaan :

$$A_{WSM} = \sum_{j=1}^{n} q_{ij} w_j , \qquad for\ i = 1, 2, 3, ..., M$$

Dengan :

A = nilai dari alternatif terbaik

n = nomor dari jumlah kriteria

q_{ij} = nilai dari i alternatif

w_j = nilai bobot

Jika persamaan di atas diterapkan pada kasus ini dengan menerapkan metode Multikriteria, maka langkah-langkah yang perlu dilakukan adalah sebagai berikut.

a. Menentukan kriteria yang menjadi dasar dalam menentukan prioritas tiap alternatifnya. Dalam kasus ini, kriteria-kriterianya adalah :

   1) Peningkatan Jumlah Mahasiswa

2) Jumlah Mahasiswa
3) Jarak Kota Tujuan
4) Jumlah SMA
5) Rata-rata IPK

b. Kemudian akan dilakukan pembobotan pada setiap kriteria dari masukan pengguna seperti yang tampak pada Tabel 2.1.

Tabel 2.1 Bobot Kriteria

Kriteria	Bobot
Peningkatan Jumlah Mahasiswa	40
Jumlah Mahasiswa	15
Jarak Kota Tujuan	25
Jumlah SMA	10
Rata-rata IPK	10

Setelah pembobotan kriteria, setiap data akan diberi nilai sesuai dengan apa yang dimasukkan oleh pengguna. Berikut ini adalah cara penilaian data dari kriteria tertentu berdasarkan masukkan pengguna.

Masukan peningkatan setiap kota adalah dalam bentuk skala angka meliputi skala awal dan skala ahir. Data penigkatan kota yang berada dalam skala yang dimasukkan oleh pengguna akan bernilai 100. Penilaian peningkatan kota terlihat pada Tabel 2.2.

Tabel 2.2 Penilaian Peningkatan Kota

Data Peningkatan Tiap Kota (%)	Masukan Peningkatan Kota oleh Pengguna (%)
	Skala Awal dan Skala Akhir
< Skala Awal	30
≥ Skala Awal dan ≤ Skala Akhir	100
≥ Skala Akhir	60

Jumlah mahasiswa berdasarkan kota asal dimasukkan dalam bentuk skala angka meliputi skala awal dan skala akhir. Data jumlah mahasiswa yang berada dalam skala angka yang dimasukan oleh pengguna akan bernilai 100 seperti yang terlihat pada Tabel 2.3.

Tabel 2.3 Penilaian Jumlah Mahasiswa

Jumlah Mhs Tiap Kota	**Masukan Jumlah Mhs oleh Pengguna**
	Skala Awal dan Skala Akhir
< Skala Awal	30
≥ Skala Awal dan ≤ Skala Akhir	100
≥ Skala Akhir	60

Jarak kota tujuan dibagi menjadi tiga ketegori yaitu lokal, dekat, jauh. Kota yang termasuk dalam kategori yang dimasukan pengguna akan bernilai 100 seperti yang terlihat pada Tabel 2.4.

Tabel 2.4 Penilaian Jarak Kota Tujuan

Jarak Kota dari Yk	**Masukan Jarak Kota oleh Pengguna**		
	Lokal	**Dekat**	**Jauh**
Lokal	100	60	30
Dekat	60	100	60
Jauh	30	30	100

Sama seperti bentuk masukan jumlah mahasiswa, jumlah SMA juga dimasukkan adalah dalam bentuk skala angka meliputi skala awal dan skala akhir. Jumlah SMA yang berada dalam skala angka yang dimasukan oleh pengguna akan bernilai 100 seperti yang terlihat pada Tabel 2.5.

Tabel 2.5 Penilaian Jumlah SMA

Jumlah SMA Tiap Kota	Masukan Jumlah SMA oleh Pengguna
	Skala Awal dan Skala Akhir
< Skala Awal	30
≥ Skala Awal dan ≤ Skala Akhir	100
≥ Skala Akhir	60

Masukkan untuk rata-rata IPK adalah dalam bentuk skala IPK. Rata-rata IPK yang berada dalam skala IPK yang dimasukan pengguna akan bernilai 100 seperti yang terlihat pada Tabel 2.6.

Tabel 2.6 Penilaian Rata-Rata IPK

Range IPK Tiap Kota	Masukan Range IPK oleh Pengguna		
	0.00 – < 2.05	2.05 – < 3.00	3.00 – 4.00
0.00 – < 2.05	100	60	30
2.05 – < 3.00	60	100	60
3.00 – 4.00	30	30	100

c. Melakukan perhitungan data penilaian kota dengan dengan persamaan WSM.
d. Melakukan pengurutan berdasarkan penilaian tertinggi.

Setelah langkah-langkah di atas dilakukan, maka akan dihasilkan perolehan nilai dari setiap kota. Data kota dengan nilai paling tinggi adalah alternatif kota yang paling mendekati keinginan pengguna.

BAB 3

ANALISIS DAN PERANCANGAN SISTEM

3.1 Metode Pengumpulan dan Analisis Data

Data dan informasi yang dibutuhkan untuk sistem ini diperoleh penulis dengan cara melakukan wawancara dengan Kepala Humas dan Admisi UKDW. Kemudian data hasil wawancara tersebut dikumpulkan dan dipelajari untuk menentukan kriteria-kriteria yang akan diterapkan pada sistem. Selain itu, data lainnya didapat penulis dari Unit Pusat Pelayanan Informasi dan Internet Kampus (PUSPINDIKA) UKDW. Data tersebut adalah data mahasiswa Fakultas Teknologi Informasi dan Fakultas Bisnis UKDW tahun 2007 – 2011. Data mahasiswa terdiri dari Nim, SMA, Kota SMA, Propinsi SMA, dan IPK. Analisis dilakukan penulis berdasarkan data-data tersebut.

Dalam penelitian ini sistem yang dibuat memiliki karakteristik sebagai berikut:

a. Batasan sistem yang dibuat yaitu untuk menentukan kota tujuan promosi bagi Fakultas Teknologi Informasi dan Fakultas Bisnis UKDW berdasarkan kriteria-kriteria yang sudah ditetapkan. Kota-kota yang direkomendasikan adalah kota-kota yang sudah terdapat dalam *database*.
b. Program yang digunakan untuk mengolah data adalah Microsoft Visual Studio 2010 dengan bahasa pemrograman C#.
c. Dalam pengolahan data untuk proses penentuan kota digunakan metode Multikriteria.
d. Keluaran sistem adalah daftar kota-kota yang ditentukan menjadi kota tujuan promosi sesuai dengan kriteria dan keinginan pengguna.

3.2 Data Flow Diagram (DFD)

3.2.1 Diagram Konteks

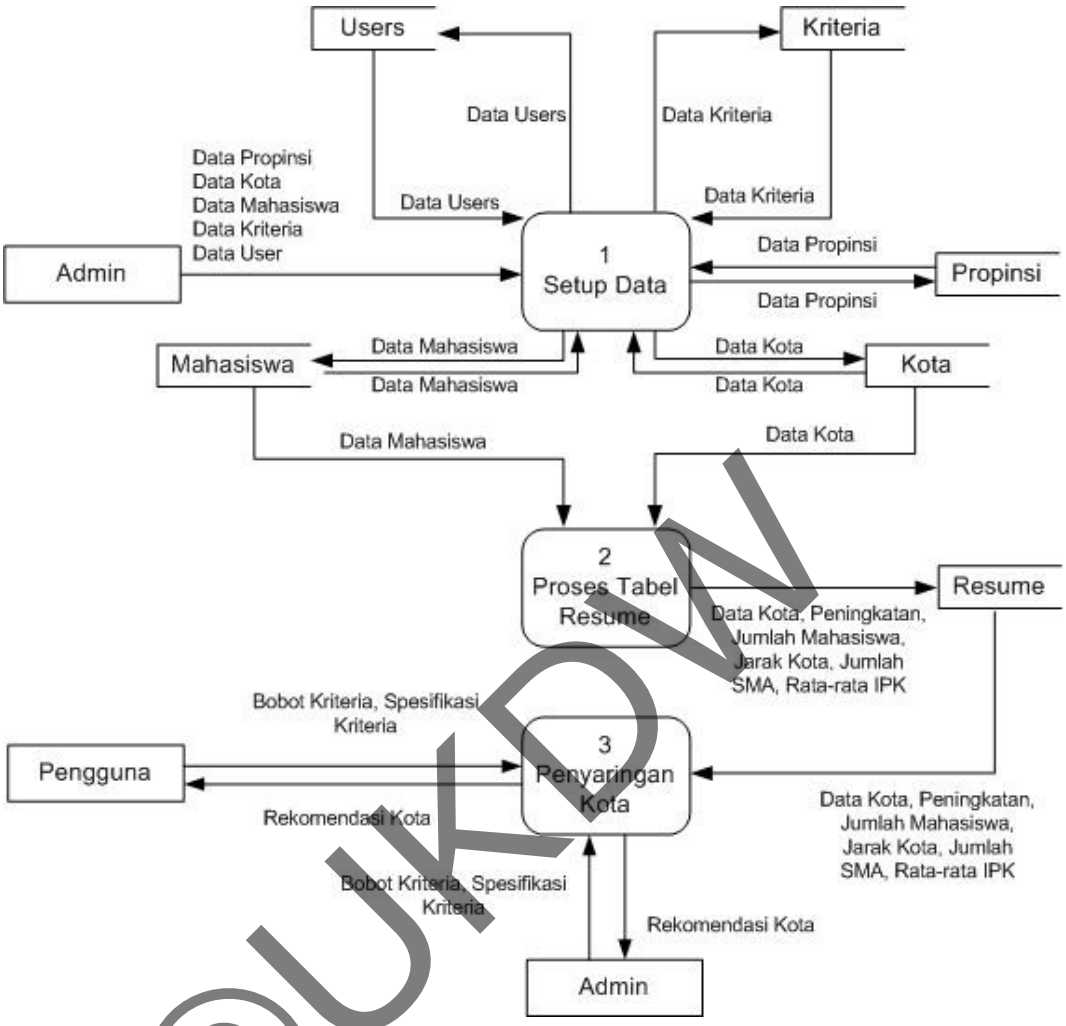
Diagram konteks adalah diagram tingkat atas yang menggambarkan aliran data yang masuk dan yang keluar dari sistem dengan kesatuan luar serta mewakili seluruh proses yang terjadi dalam sistem. Konteks diagram sistem yang dibuat, yaitu :

Gambar 3.1. Diagram Konteks

Pada Gambar 3.1 diagram konteks terdiri dari 2 (dua) entitas luar yang berinteraksi dengan sistem, yaitu admin dan pengguna. Data masukan (*input*) dari sistem berasal dari admin dan pengguna, dan data keluaran (*output*) dari sistem ditujukan kepada admin dan pengguna. Dari diagram konteks tersebut akan diuraikan menjadi bentuk DFD level 0.

3.2.2 Data Flow Diagram Tingkat 0

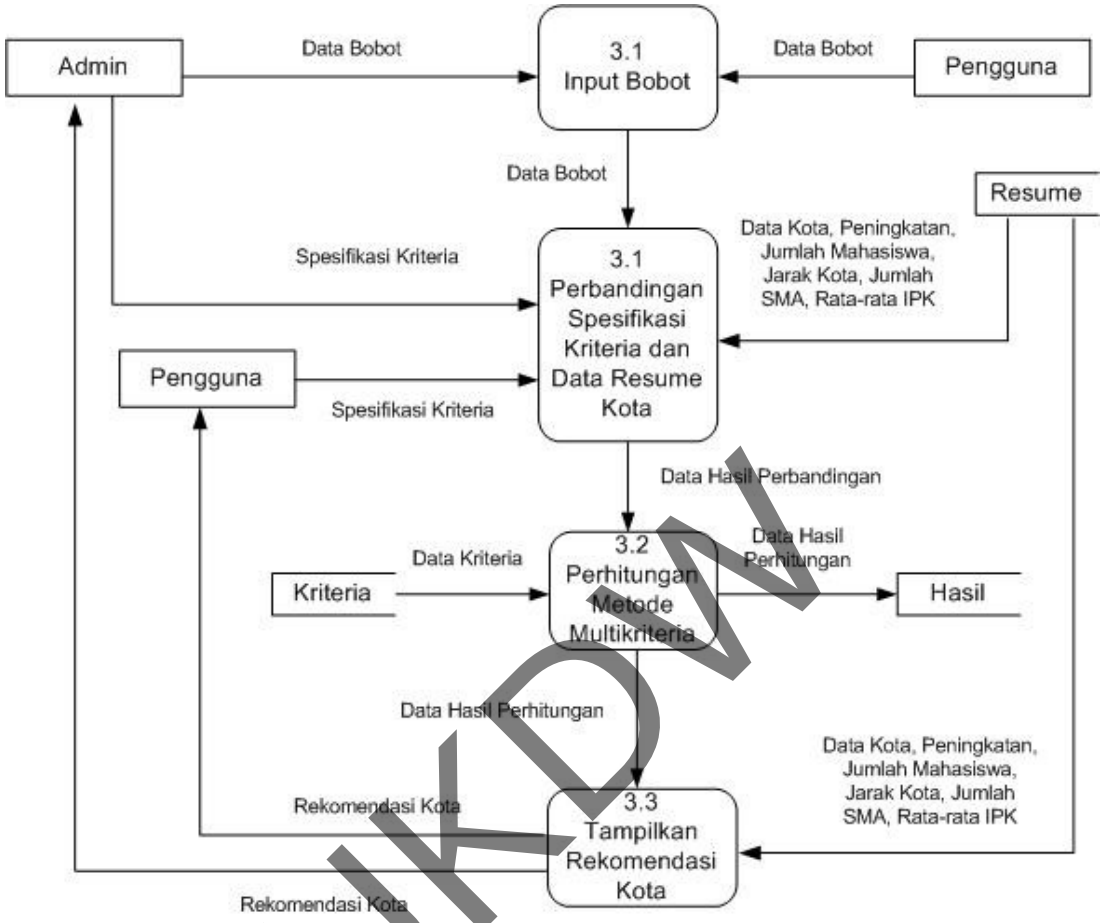
DFD tingkat 0 yaitu merupakan gambaran sistem yang lebih rinci. DFD tingkat 0 merupakan pengembangan masukan dan keluaran yang ditetapkan dalam diagram konteks menjadi empat proses yang terjadi yaitu setup data, proses tabel resume, dan penyaringan kota. Adapun entitas yang berperan dalam proses-proses tersebut adalah admin dan pengguna. Data Flow Diagram Tingkat 0 tersebut dapat dilihat pada Gambar 3.2.



Gambar 3.2. Data Flow Diagram Tingkat 0

3.2.3 Data Flow Diagram Tingkat 1

Data Flow Diagram Tingkat 1 merupakan rincian dari proses-proses yang ada pada DFD Tingkat 0. Gambar 3.3 merupakan uraian lebih lanjut dari penyaringan kota. DFD ini menggambarkan aliran data dari pengguna dan admin ke sistem, dan dari sistem ke pengguna dan admin. Proses yang terdapat dalam DFD ini meliputi proses perbandingan spesifikasi kriteria dan data resume kota, perhitungan metode Multikriteria, dan menampilkan rekomendasi kota.



Gambar 3.3 Data Flow Diagram Tingkat 1 Penyaringan Kota

3.3 Kamus Data

Data yang digunakan disimpan dalam tabel, keterangan mengenai *field* yang digunakan untuk menyimpan data didefinisikan dalam Tabel 3.1.

Tabel 3.1 Kamus Data

Nama Field	Tipe Data	Aturan	Keterangan
kd_kota	char(6)	Not Null	Kode kota
nm_kota	varchar(50)	Not Null	Nama kota
jenis	varchar(10)	Not Null	Jenis kota
kd_prop	char(3)	Not Null	Kode propinsi
nm_prop	varchar(50)	Not Null	Nama propinsi
jarak	varchar(20)	Not Null	Jarak Propinsi
nim	char(8)	Not Null	Nim mahasiswa

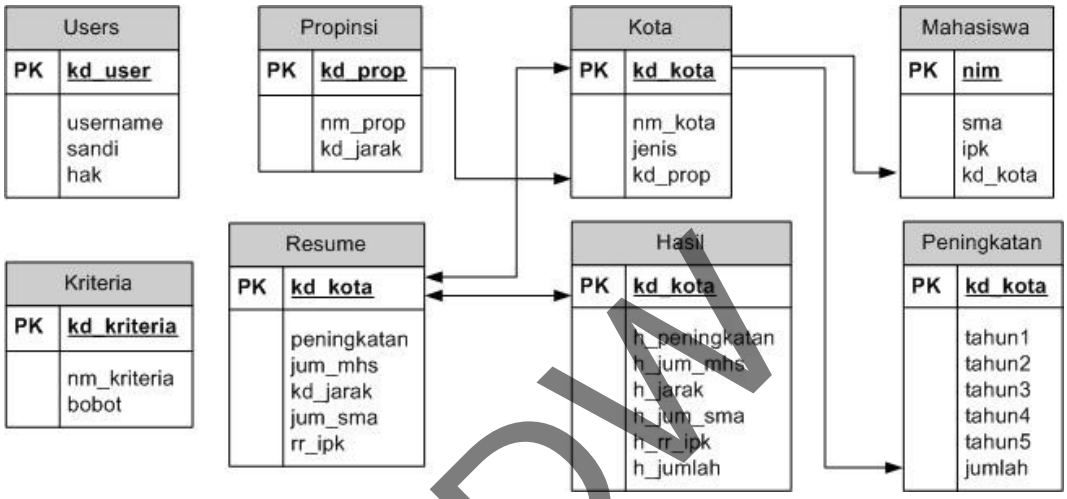
Tabel 3.1 Kamus Data (lanjutan)

Nama Field	Tipe Data	Aturan	Keterangan
sma	varchar(100)	Not Null	SMA mahasiswa
ipk	float	Not Null	IPK mahasiswa
kd_user	char(2)	Not Null	Kode user
username	varchar(50)	Not Null	Username
sandi	varchar(15)	Not Null	Sandi user
hak	varchar(15)	Not Null	Hak user
kd_kriteria	int	Not Null	Kode kriteria
nm_kriteria	varchar(50)	Not Null	Nama kriteria
bobot	int	Not Null	Bobot kriteria
tahun1	int	Not Null	Jumlah mahasiswa tahun 1
tahun2	int	Not Null	Jumlah mahasiswa tahun 2
tahun3	int	Not Null	Jumlah mahasiswa tahun 3
tahun4	int	Not Null	Jumlah mahasiswa tahun 4
tahun5	int	Not Null	Jumlah mahasiswa tahun 5
jumlah	float	Not Null	Jumlah mahasiswa 5 tahun
rr_naik	float	Not Null	Rata-rata nilai naik
rr_turun	float	Not Null	Rata-rata nilai turun
peningkatan	float	Not Null	Peningkatan mahasiswa tiap kota
jum_mhs	int	Not Null	Jumlah mahasiswa tiap kota
jum_sma	int	Not Null	Jumlah SMA tiap kota
rr_ipk	float	Not Null	Rata-rata IPK tiap kota
h_peningkatan	int	Not Null	Hasil peningkatan
h_jum_mhs	int	Not Null	Hasil jumlah mahasiswa
h_jarak	int	Not Null	Hasil jarak
h_jum_sma	int	Not Null	Hasil jumlah SMA
h_rr_ipk	int	Not Null	Hasil rata-rata IPK
h_jumlah	int	Not Null	Jumlah hasil

Sistem ini mempunyai 8 tabel, yaitu propinsi, kota, mahasiswa, peningkatan, resume, hasil, user, dan kriteria. Relasi antar tabel terjadi antara tabel propinsi dan kota, kota dan mahasiswa, kota dan peningkatan, kota dan resume, resume dan hasil.

3.4 Perancangan Database

Perancangan database yang digunakan dalam sistem dapat dilihat pada Gambar 3.4.

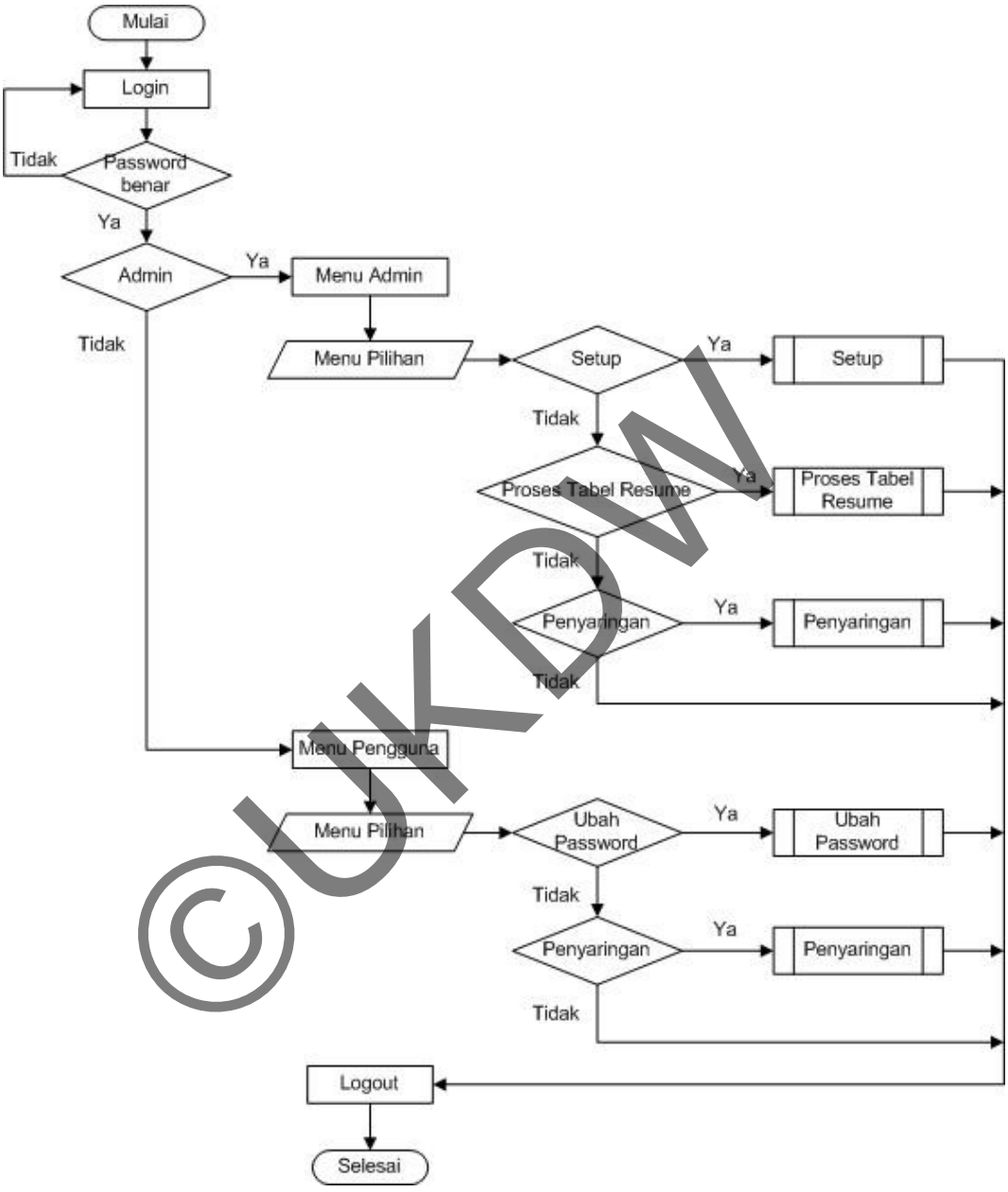


Gambar 3.4 Perancangan Database

3.5 Alur Proses Kerja Sistem

Alur proses kerja sistem yang akan dibangun dapat dilihat melalui *flowchart*. *Flowchart* adalah bentuk gambar atau diagram yang mempunyai aliran satu atau dua arah secara sekuensial. Selain itu *flowchart* juga dapat digunakan untuk mempresentasikan ataupun merancang program.

3.5.1 *Flowchart* Sistem



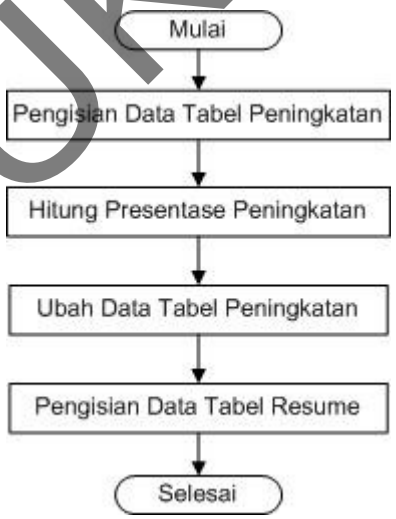
Gambar 3.5 *Flowchart* Sistem

Dalam proses perancangan sistem, akan digambarkan alur kerja sistem keseluruhan dalam bentuk *flowchart*. Alur kerja sistem merupakan algoritma dari jalannya sistem yang dimulai dengan proses *login* yang bertujuan untuk membedakan admin atau pengguna. Admin memiliki hak akses untuk setup, proses tabel resume, dan penyaringan kota, sedangkan pengguna hanya memiliki

hak akses untuk penyaringan kota dan ubah *password*. Setelah login maka pengguna sistem dapat mengakses menu pilihan sesuai dengan hak akses masing-masing. Gambar 3.5 di atas adalah perancangan alur kerja sistem kelseluruhan mulai dari proses *login* untuk masuk ke sistem, menu pilihan sesuai dengan hak akses masing-masing pengguna sistem, dan terakhir proses *logout* atau keluar dari sistem.

3.5.2 *Flowchart* Proses Tabel Resume

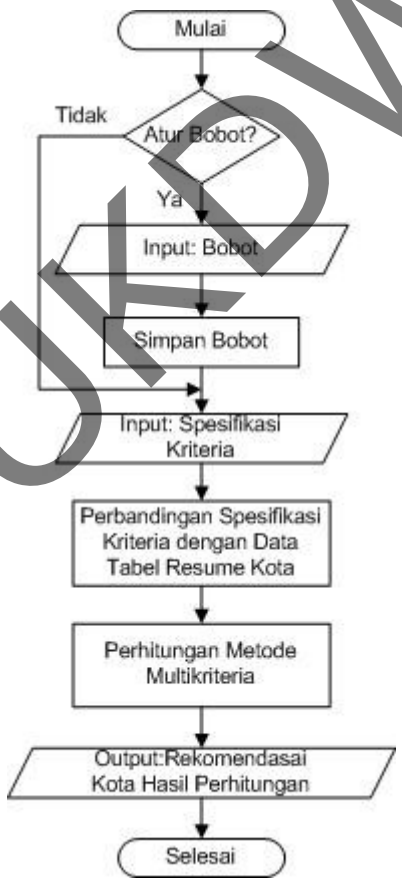
Gambar 3.6 adalah gambar alur proses tabel Resume. Pada proses ini pengisian tabel Resume dimulai dengan pengisian data pada tabel Peningkatan. Kemudian sistem melakukan proses perhitungan presentase peningkatan pada setiap kota. Hasil perhitungan kemudian dimasukan dalam tabel Peningkatan melalui proses ubah data tabel peningkatan. Setelah itu, sistem akan mengisi data pada tabel Resume berdasarkan data pada tabel Kota, Mahasiswa, dan Peningkatan.



Gamba 3.6 *Flowchart* Proses Tabel Resume

3.5.3 *Flowchart* Penyaringan Kota

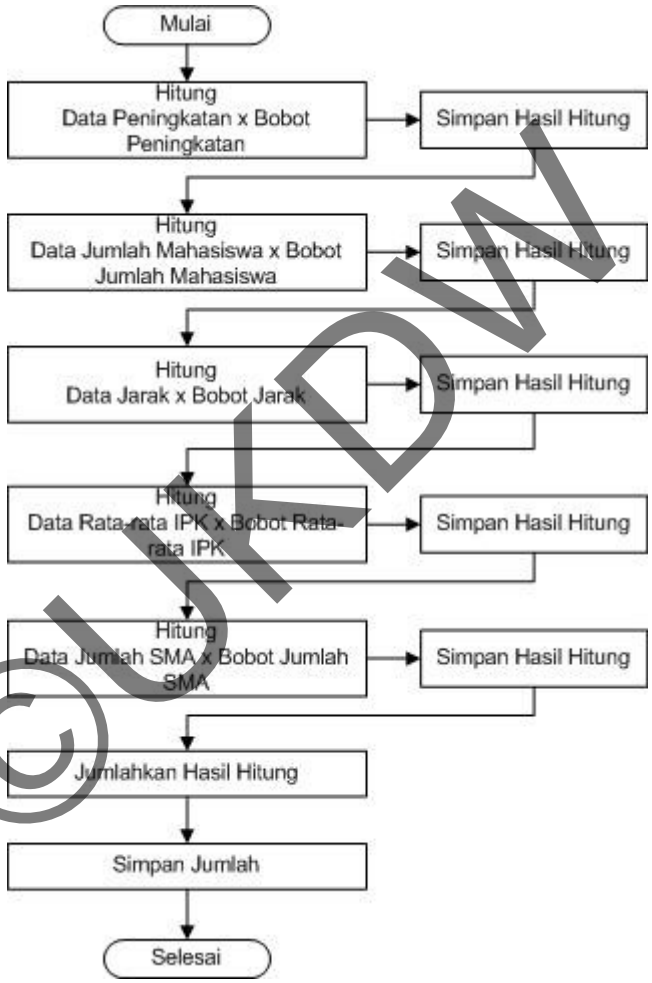
Gambar 3.7 adalah gambar alur proses penyaringan kota. Pada proses ini pengguna memasukkan bobot dan spesifikasi keriteria. Sistem akan melakukan perbandingan spesifikasi kriteria yang dimasukkan dengan data pada Tabel Resume. Perbandingan akan menghasilkan nilai. Nilai tersebut akan di gunakan oleh sistem untuk melakukan proses perhitungan dengan Metode Multikriteria. Perhitungan ini juga melibatkan bobot kriteria yang sebelumnya dimasukkan oleh pengguna. Perhitungan kemudian akan menghasilkan rekomendasi kota sesuai dengan hasil perhitungan.



Gambar 3.7 *Flowchart* Penyaringan Kota

3.5.4 *Flowchart* Perhitungan Metode Multikriteria

Gambar 3.8 adalah gambar alur perhitungan dengan metode Multikriteria. Setiap data dikalikan dengan bobot dari data tersebut, kemudian hasil perkalian akan disimpan oleh sistem. Setelah perhitungan setiap data selesai dilakukan, sistem akan menjumlahkan hasil perhitungan.



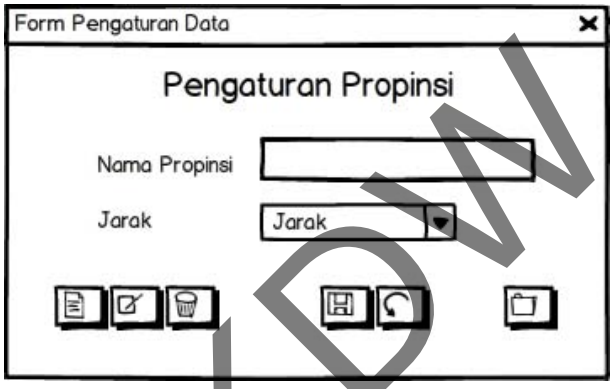
Gambar 3.8 Flowchart Perhitungan Metode Multikriteria

3.6 Perancangan Antarmuka Pengaturan Data

Pengaturan data merupakan tahap penyiapan data sebelum data diolah untuk dianalisis dan digunakan pada proses-proses selanjutnya. Pengaturan data hanya dilakukan oleh Admin. Pengaturan data meliputi pengaturan propinsi,

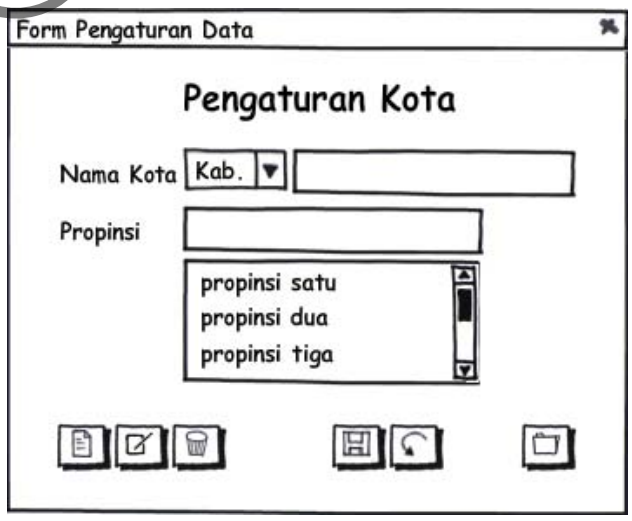
pengaturan kota, pengaturan mahasiswa, pengaturan bobot kriteria, dan pengaturan pengguna. Berikut ini adalah penjelasan mengenai beberapa Pengaturan Data tersebut.

Pengaturan propinsi bertujuan untuk menambah, mengubah, dan menghapus data propinsi. Data yang perlu dimasukkan untuk menambah propinsi adalah nama propinsi dan jarak propinsi tersebut dari kota Yogyakarta. *Form* pengaturan propinsi terlihat pada Gambar 3.9.



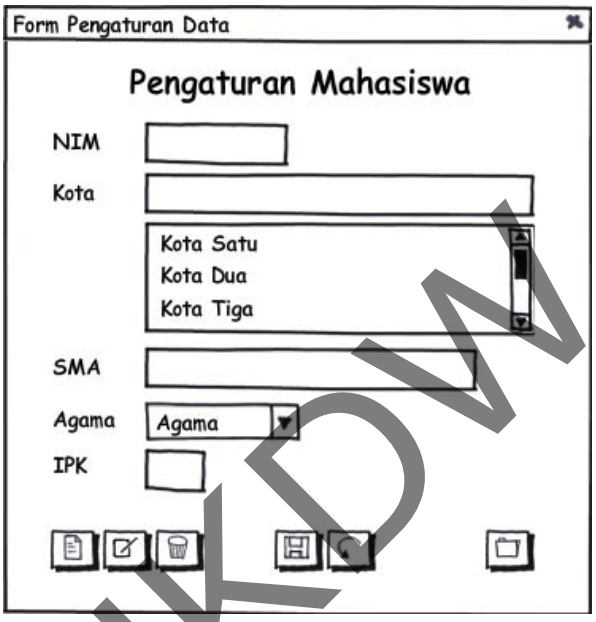
Gambar 3.9 Form Pengaturan Propinsi

Pengaturan kota bertujuan untuk menambah, mengubah, dan menghapus data kota. Data yang perlu dimasukkan untuk melakukan penambahan kota adalah jenis kota yang terdiri dari kabupaten dan kota madya, nama kota, dan propinsi kota tersebut. *Form* pengaturan kota terlihat pada Gambar 3.10.



Gambar 3.10 Form Pengaturan Kota

Pengaturan mahasiswa bertujuan untuk menambah, mengubah dan menghapus data mahasiswa. Data yang perlu dimasukkan untuk melakukan penambahan mahasiswa adalah NIM, kota asal SMA, nama SMA, agama, dan IPK. *Form* pengaturan mahasiswa telihat pada Gambar 3.11.

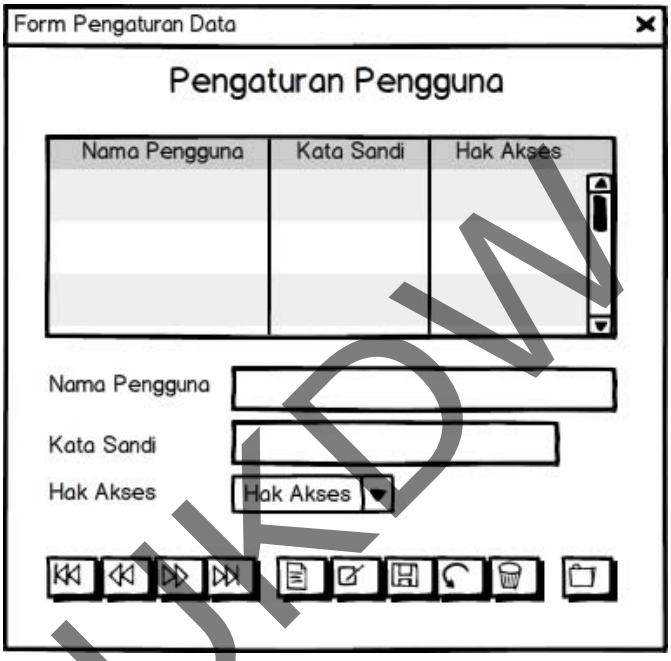


Gambar 3.11 Form Pengaturan Mahasiswa

Pengaturan bobot kriteria bertujuan untuk mengubah bobot setiap kriteria.. *Form* Pengaturan Kriteria terlihat pada Gambar 3.12.

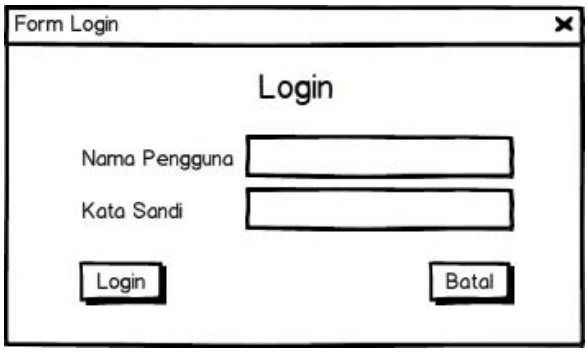
Gambar 3.12 Form Pengaturan Bobot Kriteria

Pengaturan pengguna bertujuan untuk menambah, mengubah dan menghapus data pengguna sistem. Data yang perlu dimasukkan untuk melakukan penambahan pengguna adalah nama pengguna, sandi dan hak akses. Hak akses terdiri atas admin dan pengguna. *Form* Pengaturan Pengguna terlihat pada Gambar 3.13.



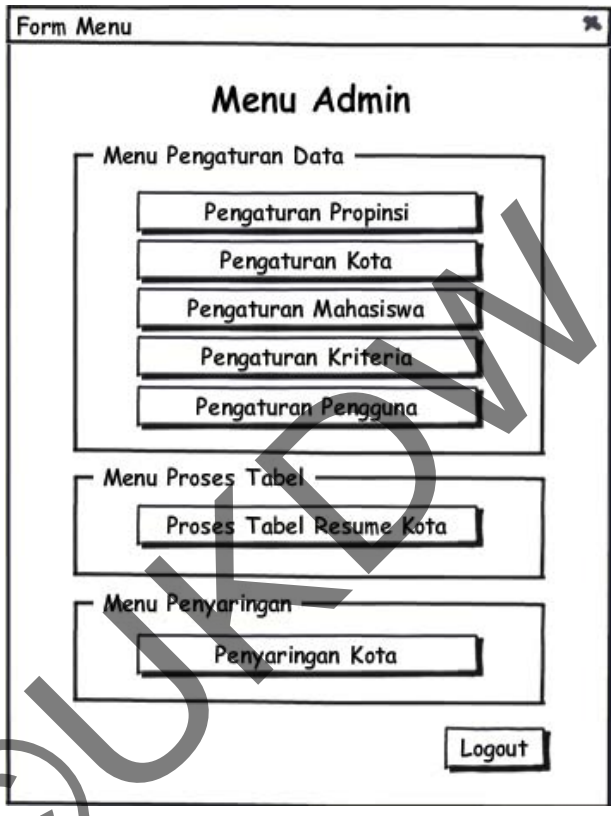
Gambar 3.13 Form Pengaturan Pengguna

Hak akses digunakan untuk membedakan Form yang dapat diakses oleh setiap pengguna. Oleh karena itu, setiap pengguna perlu melakukan Login terlebih dahulu ke dalam sistem sehingga sistem dapat membedakan hak akses setiap pengguna sesuai data yang ada. *Form Login* terlihat pada Gambar 3.14.

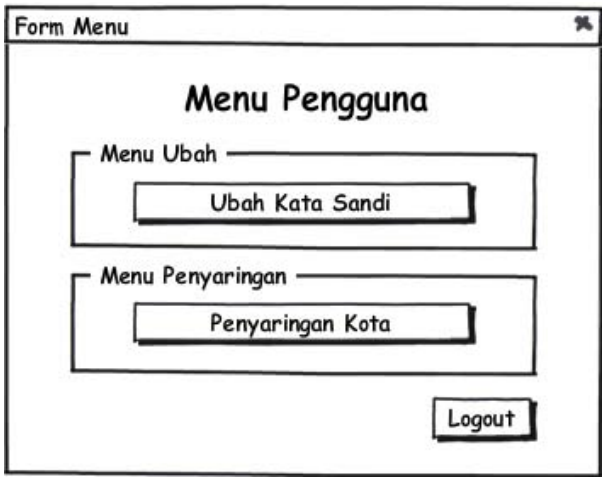


Gambar 3.14 Form *Login*

Gambar 3.15 merupakan Form Menu Admin yang tampil jika pengguna login dengan hak akses sebagai admin sedangkan Gambar 3.16 merupakan Form Menu Pengguna yang tampil jika pengguna login dengan hak akses sebagai pengguna biasa.



Gambar 3.15 Form Menu Admin

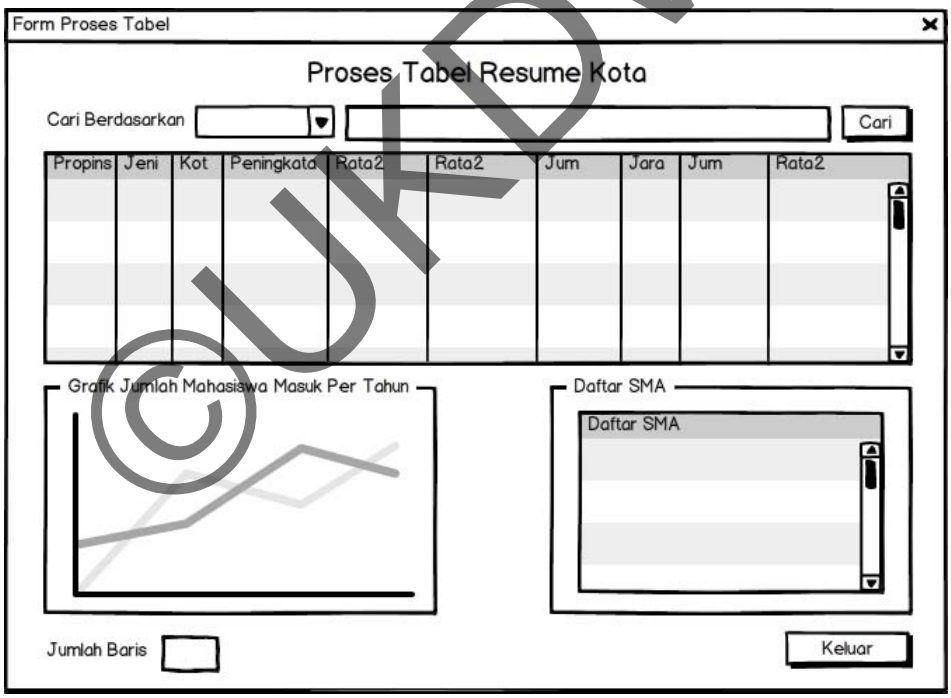


Gambar 3.16 Form Menu Pengguna

3.7 Perancangan Antarmuka Proses Sistem

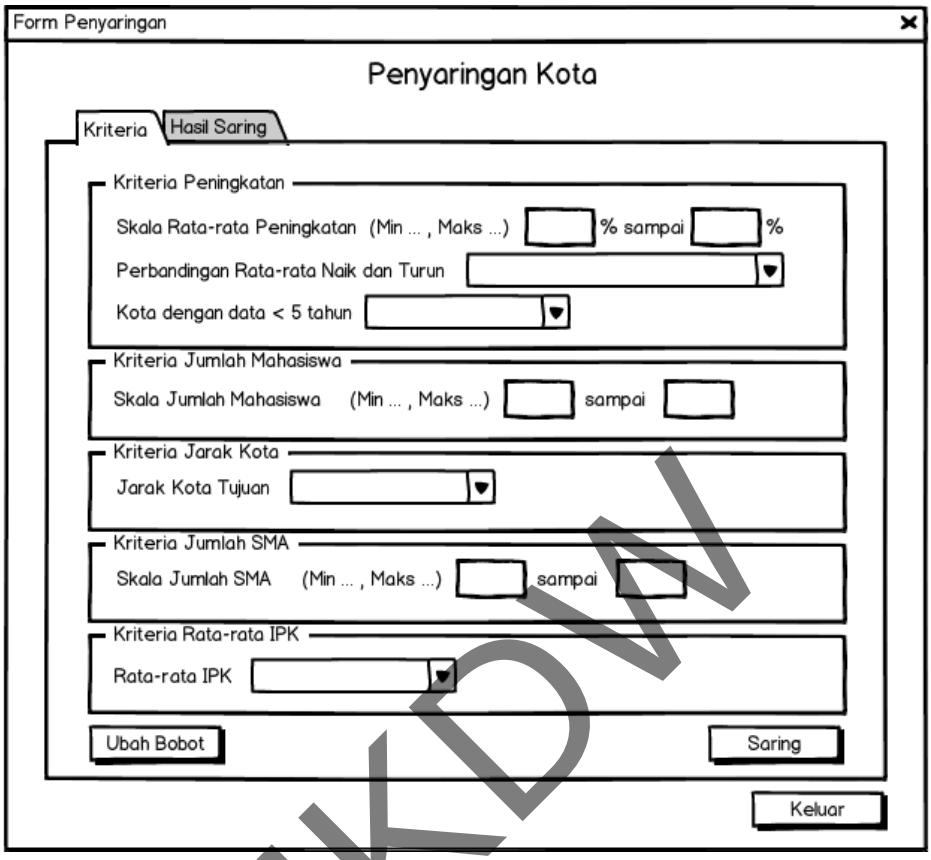
Proses merupakan tahap inti yang dilakukan oleh pengguna sistem. Proses terdiri dari dua bagian, yaitu proses untuk mengisi Tabel Resume Kota dan proses Penyaringan Kota. Berikut ini adalah penjelasan mengenai beberapa proses tersebut.

Proses Tabel Resume Kota bertujuan untuk mengisi Tabel Resume Kota berdasarkan data-data yang sudah dimasukkan atau sudah diatur sebelumnya oleh Admin. Tabel Resume adalah tabel yang berisikan data-data yang berkaitan dengan kota yang diperlukkan untuk proses penyaringan kota. Proses Tabel Resume Kota ini hanya dapat digunakan oleh Admin. *Form* Proses Tabel Resume Kota terlihat pada Gambar 3.17.



Gambar 3.17 Form Proses Tabel Resume Kota

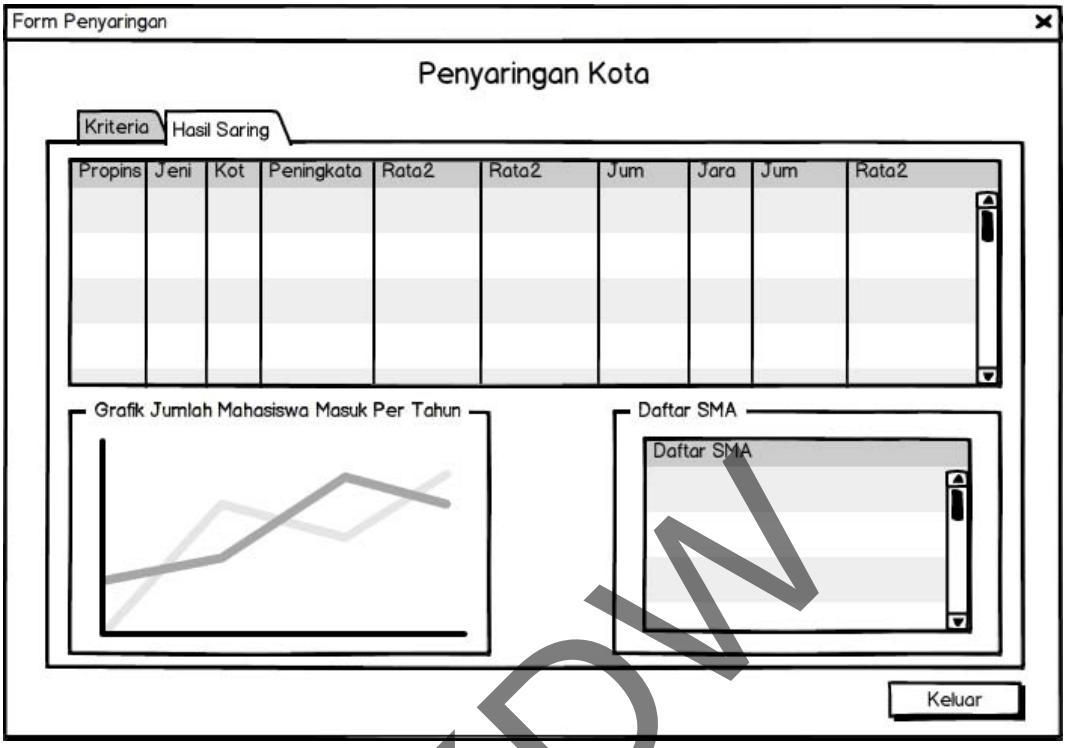
Proses Penyaringan Kota bertujuan untuk menyaring kota berdasarkan nilai kriteria yang ditentukan. Kriteria-kriteria tersebut meliputi peningkatan, jumlah mahasiswa, jarak kota tujuan, jumlah SMA dan rata-rata IPK. *Form* Penyaringan Kota terlihat pada Gambar 3.18.



Gambar 3.18 Form Penyaringan Kota

3.8 Perancangan Antarmuka Hasil Sistem

Hasil sistem merupakan tahap akhir yang dilakukan di dalam sistem untuk menampilkan hasil penyaringan kota. Penyaringan kota menghasilkan beberapa kota yang sesuai dengan kriteria yang dimasukkan. Hasil penyaringan ditampilkan menggunakan *Data Grid View* dan *Line Chart*. Form Hasil Penyaringan terlihat pada Gambar 3.19.



Gambar 3.19 *Form* Hasil Penyaringan Kota

BAB 4

PENERAPAN DAN ANALISIS SISTEM

Bab 4 berisi penerapan dan analisis sistem yang mengacu pada perancangan yang dibahas pada Bab 3. Sistem diterapkan dengan menggunakan *software Microsoft Visual Studio* dan bahasa pemrograman C#.

4.1 Penerapan Sistem

4.1.1 Sistem Autentifikasi

Sistem autentifikasi digunakan untuk mengautentifikasi setiap pengguna yang hendak masuk ke dalam sistem. Hal ini bertujuan untuk membatasi fitur atau halaman yang dapat diakses oleh pengguna sesuai dengan hak aksesnya. Proses autentifikasi dilakukan melalui *form Login* seperti yang terlihat pada Gambar 4.1. Pada *form Login*, sistem akan mencocokan nama pengguna dan kata sandi yang diinputkan oleh pengguna dengan data pengguna yang terdapat dalam *database* sistem. Jika nama pengguna dan kata sandi tidak cocok, maka sistem akan mengeluarkan pesan kesalahan *login*.

Gambar 4.1 Form *Login*

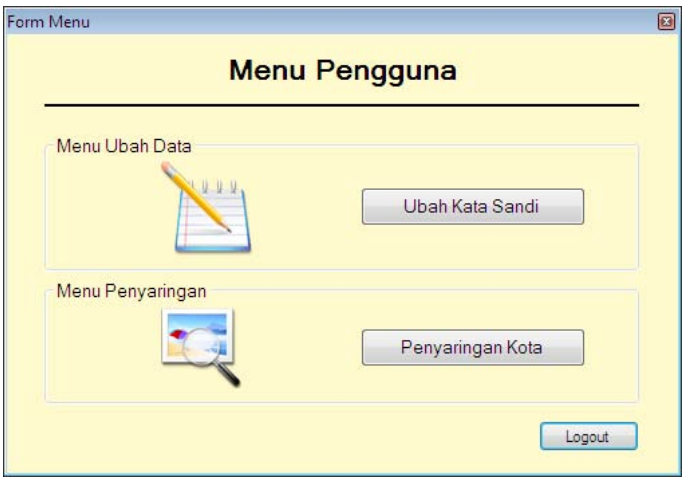
4.1.2 Menu Sistem

Menu sistem dibagi menjadi dua bagian, yaitu menu admin dan menu pengguna. Kedua menu tersebut menyediakan fitur-fitur yang berbeda. Di dalam

menu admin, terdapat fitur untuk mengatur data, memproses tabel, dan melakukan penyaringan kota seperti terlihat pada Gambar 4.2.

Gambar 4.2 Form Menu Admin

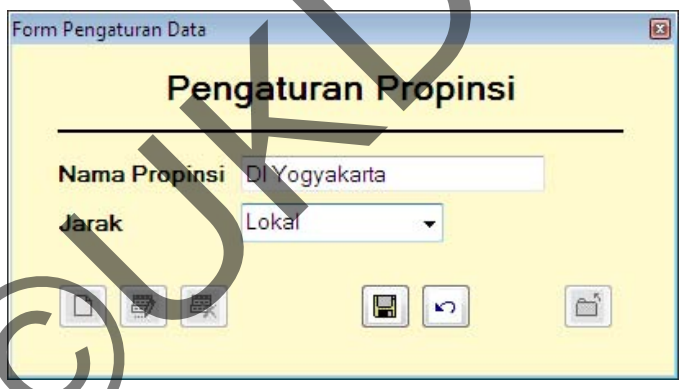
Admin memiliki fitur lebih banyak dibanding dengan pengguna biasa. Menu pengguna hanya menyediakan fitur untuk mengubah kata sandi dan melakukan penyaringan kota seperti terlihat pada Gambar 4.3.



Gambar 4.3 Form Menu Pengguna

4.1.3 Pengaturan Data

Form pengaturan data terdiri dari lima bagian, yaitu pengaturan propinsi, kota, mahasiswa, kriteria, dan pengguna. Pada *form* pengaturan propinsi, admin dapat melakukan penambahan, pengubahan, serta pengahapusan data propinsi. Pada Gambar 4.4 terdapat tombol untuk manipulasi data, dan tombol keluar. Terdapat juga tombol simpan dan batal yang hanya akan berfungsi ketika admin melakukan manipulasi data seperti menambah, mengubah dan menghapus data. Untuk menambah propinsi, admin perlu mengisi nama propinsi, dan memilih jarak yang sesuai dengan propinsi tersebut. Untuk mengubah dan menghapus propinsi admin terlebih dahulu perlu memilih propinsi yang hendak diubah ataupun dihapus. Pemilihan tersebut dilakukan pada *form* pencarian propinsi yang akan muncul ketika admin menekan tombol ubah atau tombol hapus.



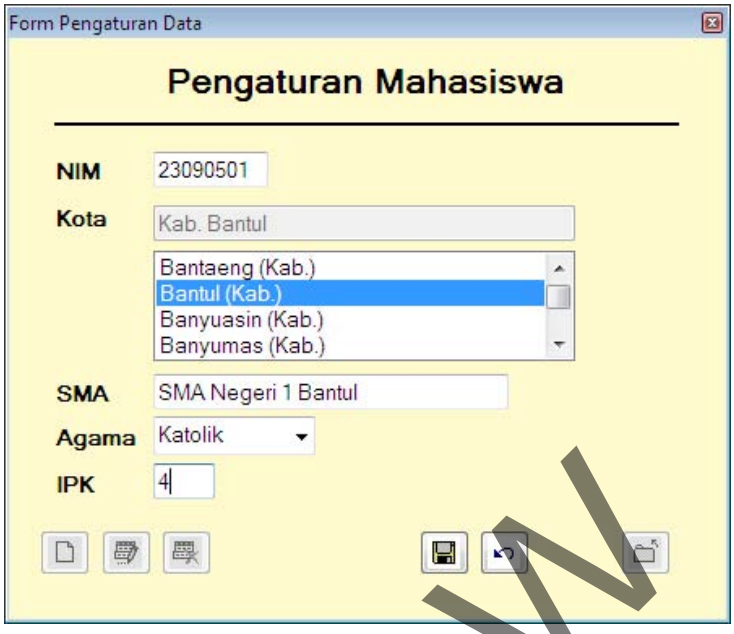
Gambar 4.4 Form Pengaturan Propinsi

Pada *form* pengaturan kota, admin dapat melakukan penambahan, pengubahan, serta pengahapusan data kota. Pada Gambar 4.5 terdapat tombol untuk manipulasi data, dan tombol keluar. Terdapat juga tombol simpan dan batal yang hanya akan berfungsi ketika admin melakukan manipulasi data seperti menambah, mengubah dan menghapus data. Untuk menambah kota, admin perlu memilih jenis kota yang dibagi menjadi 2 bagian yaitu kabupaten dan kota madya, mengisi nama kota, dan memilih propinsi kota tersebut. Untuk mengubah dan menghapus kota, admin terlebih dahulu perlu memilih kota yang hendak diubah

ataupun dihapus. Pemilihan tersebut dilakukan pada *form* pencarian kota yang akan muncul ketika admin menekan tombol ubah atau tombol hapus.

Gambar 4.5 Form Pengaturan Kota

Pada *form* pengaturan mahasiswa, admin dapat melakukan penambahan, pengubahan, serta pengahapusan data mahasiswa. Pada Gambar 4.6 terdapat tombol untuk manipulasi data, dan tombol keluar. Terdapat juga tombol simpan dan batal yang hanya akan berfungsi ketika admin melakukan manipulasi data seperti menambah, mengubah dan menghapus data. Untuk menambah mahasiswa, admin perlu mengisi NIM, memilih kota asal SMA, mengisi nama SMA, memilih agama, dan mengisi IPK. Pemilihan kota asal dilakukan melalui *form* pencarian kota yang akan muncul ketika admin menekan tombol dengan tulisan "…". Pilihan agama yang tersedia yaitu Kristen, Katolik, Islam, Hindu dan Budha. Untuk mengubah dan menghapus mahasiswa, admin terlebih dahulu perlu memilih mahasiswa yang hendak diubah ataupun dihapus. Pemilihan tersebut dilakukan pada form pencarian mahasiswa yang akan muncul ketika admin menekan tombol ubah atau tombol hapus.



Gambar 4.6 Form Pengaturan Mahasiswa

Pada *form* pengaturan bobot kriteria, admin hanya dapat melakukan pengubahan bobot. Pada Gambar 4.7 pengubahan bobot dapat dilakukan melaui tombol ubah. Bobot kriteria diisi dalam bentuk angka dengan nilai minimal 0 dan maksimal 100. Jika total bobot telah melebihi nilai maksimal, maka sistem akan memberikan peringatan dalam bentuk *messagebox*.

Gambar 4.7 Form Pengaturan Bobot Kriteria

Pada *form* pengaturan pengguna, admin dapat melakukan penambahan, pengubahan, serta pengahapusan data kriteria. Pada Gambar 4.8 terdapat beberapa tombol untuk navigasi, manipulasi data, dan tombol keluar. Terdapat juga tombol simpan dan batal yang hanya akan berfungsi ketika admin melakukan manipulasi data seperti menambah, mengubah dan menghapus data. Tombol navigasi meliputi tombol *top*, tombol *previous*, tombol *next*, dan *end*. Tombol navigasi berfungsi untuk mengarahkan *record pointer* agar bergeser ke baris paling depan, baris sebelumnya, baris berikutnya maupun baris paling belakang sesuai dengan tombol navigasi yang ditekan oleh admin. Untuk menambah pengguna, admin perlu mengisi nama pengguna, kata sandi dan memilih hak akses yang terbagi menjadi dua bagian yaitu admin dan pengguna. Untuk mengubah dan menghapus data pengguna admin terlebih dahulu perlu memilih data pengguna yang hendak diubah ataupun dihapus.

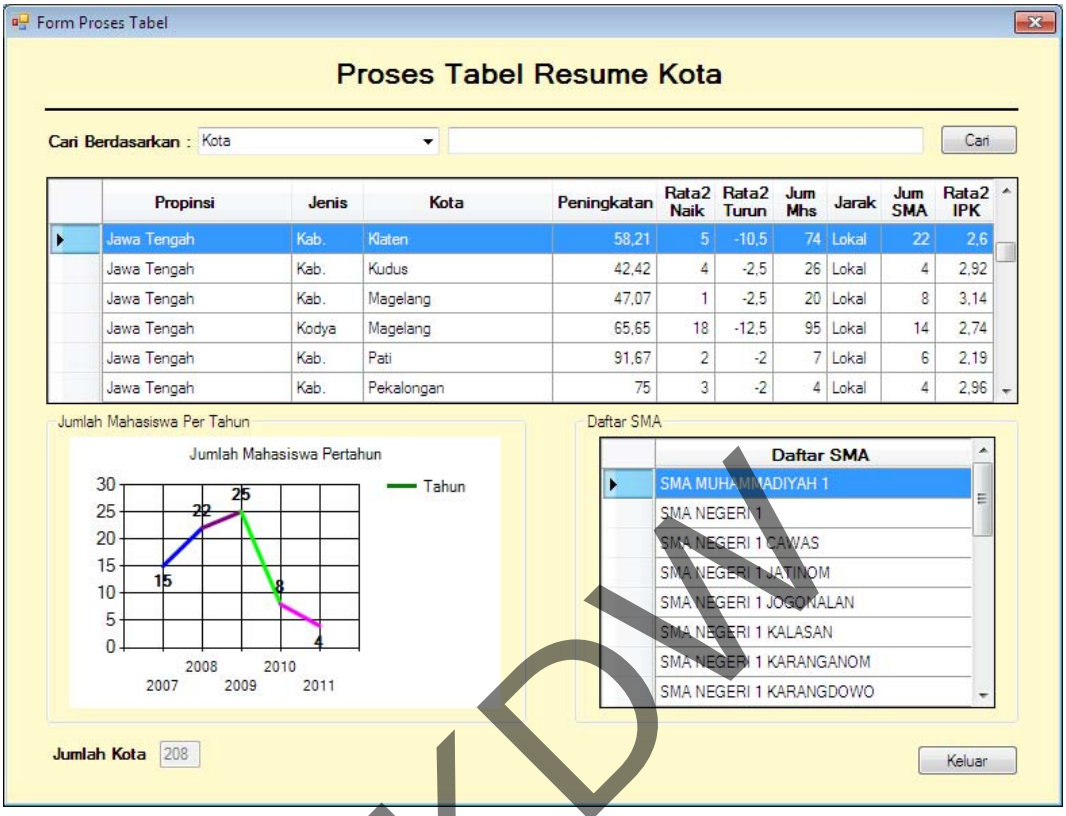
Gambar 4.8 Form Pengaturan Pengguna

4.1.4 Proses Tabel Resume Kota

Proses tabel resume kota terlihat pada Gambar 4.9 Ketika *form* ini dijalankan oleh admin, maka secara otomatis sistem akan menjalankan proses untuk mengisi tabel resume. Proses tersebut meliputi:

a. proses menghitung rata-rata presentase peningkatan jumlah mahasiswa dari setiap kota dalam rentang tahun 2007 sampai tahun 2011;
b. proses menghitung rata-rata jumlah kenaikan dan jumlah penurunan mahasiswa baru yang masuk dalam rentang tahun 2007 sampai tahun 2011;
c. proses menghitung jumlah mahasiswa yang berasal dari kota tertentu;
d. proses menghitung jumlah SMA yang terdapat pada kota tertentu sesuai dengan data asal SMA mahasiswa yang terdapat dalam *database* sistem;
e. Proses menghitung rata-rata IPK mahasiswa yang berasal dari kota tertentu.

Pengisian tabel resume kota dilakukan berdasarkan data kota, data propinsi, dan data-data hasil proses-proses yang telah disebutkan diatas. Resume kota kemudian akan ditampilkan dalam sebuah *grid view*. Jumlah mahasiswa dari kota tertentu yang masuk pada rentang tahun 2007 sampai 2012 ditampilkan dalam bentuk *line chart* sedangkan daftar SMA yang berasal dari kota tertentu ditampilkan dalam sebuah *grid view*. *Form* proses tabel resume kota juga menyediakan fitur untuk melakukan pencarian berdasarkan kota dan propinsi. Selain itu, terdapat juga pencarian dengan pilihan perbandingan yaitu rata-rata naik > rata-rata turun, rata-rata naik < rata-rata turun, dan rata-rata naik = rata-rata turun.



Gambar 4.9 Form Proses Tabel Resume Kota

4.1.5 Penyaringan Kota

Form penyaringan kota pada Gambar 3.10 bertujuan untuk menyaring kota berdasarkan kriteria yang dimasukkan oleh pengguna. Kriteria tersebut meliputi kriteria peningkatan, jumlah mahasiswa, jarak kota tujuan, jumlah SMA, dan rata-rata IPK. Masukan untuk kriteria peningkatan adalah dalam bentuk skala angka. Pada kriteria peningkatan, pengguna juga dapat memilih perbandingan rata-rata meliputi rata-rata naik > rata-rata turun, rata-rata naik < rata-rata turun, dan rata-rata naik = rata-rata turun. Selain itu, pengguna juga dapat memilih apakah kota dengan data < 5 tahun juga akan proses atau tidak. Masukan untuk kriteria jumlah mahasiswa adalah dalam bentuk skala angka. Masukan untuk jarak kota tujuan adalah dalam bentuk pilihan. Pengguna dapat memilih jarak sesuai dengan yang diinginkan untuk diproses. Masukan untuk kriteria jumlah SMA adalah dalam bentuk skala angka. Masukan untk kriteria rata-rata IPK adalah dalam pilihan.

Pada form, sistem juga menampilkan minimal dan maksimal angka yang dapat dimasukkan oleh pengguna sesuai dengan data yang terdapat dalam database.

Proses penyaringan dapat dilakukan oleh pengguna dengan menekan tombol saring sedangkan jika pengguna ingin mengubah bobot dari setiap kriteria, pengguna dapat menekan tombol ubah bobot.

Gambar 4.10 Form Penyaringan Kota (Kriteria)

Ketika pengguna melakukan penyaringan dengan menekan tombol saring, maka sistem akan melakukan perbadingan kriteria yang dimasukkan pengguna dengan data kota di Tabel Resume. Perbandingan akan dimulai dengan membandingkan rata-rata peningkatan kota yang dimasukkan oleh pengguna dengan data di *database*, kemudian sistem akan memberi nilai pada setiap data peningkatan seperti yang terlihat pada Kode Programa 4.1. Setelah data peningkatan diberi nilai, sistem akan mengalikan nilai tersebut dengan bobot peningkatan. Cara mendapatkan bobot terlihat pada Kode Program 4.2.

```
private void Peningkatan()
{
    double bobotx = GetBobot(1);
    double rangeawal = 0; double rangeakhir = 0;
    int nilai1 = 30; int nilai2 = 100; int nilai3 = 60;
    rangeawal = Convert.ToDouble(txtAwalPen.Text.Trim());
    rangeakhir= Convert.ToDouble(txtAkhirPen.Text.Trim());
    string strSql = @"update hasil set peningkatan=
                    case when(kd_kota in (select kd_kota from resume where
                            peningkatan < @range_awal))
                            then @nilai1 * @bobot
                        when(kd_kota in (select kd_kota from resume where
                            peningkatan >= @range_awal and peningkatan <= @range_akhir))
                            then @nilai2 * @bobot
                        else @nilai3 * @bobot end";
    cmd = new SqlCommand(strSql, conn);
    cmd.Parameters.AddWithValue("@range_awal", rangeawal);
    cmd.Parameters.AddWithValue("@range_akhir", rangeakhir);
    cmd.Parameters.AddWithValue("@nilai1", nilai1);
    cmd.Parameters.AddWithValue("@nilai2", nilai2);
    cmd.Parameters.AddWithValue("@nilai3", nilai3);
    cmd.Parameters.AddWithValue("@bobot", bobotx);
    conn.Close();
    conn.Open();
    cmd.ExecuteNonQuery();
}
```

Kode Program 4.1 Cara Penilaian Data Peningkatan

```
private double GetBobot(int kriteria)
{
    double bobotx = 0;
    string strSql = @"select bobot from kriteria where kd_kriteria=@kd_kriteria";
    cmd = new SqlCommand(strSql, conn);
    cmd.Parameters.AddWithValue("@kd_kriteria", kriteria);
    conn.Close();
    conn.Open();
    dr = cmd.ExecuteReader();
    dr.Read();
    bobotx = Convert.ToDouble(dr["bobot"])/100;
    return bobotx;
}
```

Kode Program 4.2 Cara Mendapatkan Bobot Kriteria

Setelah data peningkatan, sistem akan membandingkan jumlah mahasiswa yang dimasukkan oleh pengguna dengan data di *database*, kemudian sistem akan memberi nilai pada setiap data jumlah mahasiswa seperti yang terlihat pada Kode Program 4.3. Pemberian nilai dilanjutkan dengan mengalikan nilai dengan bobot kriteria jumlah mahasiswa.

```
private void JumlahMhs()
{
    double bobotx = GetBobot(2);
    int rangeawal = Convert.ToInt32(txtJumMhsAwal.Text.Trim());
    int rangeakhir = Convert.ToInt32(txtJumMhsAkhir.Text.Trim());
    int nilai1 = 30; int nilai2 = 100; int nilai3 = 60;
    string strSql = @"update hasil set jum_mhs=
                    case when(kd_kota in (select kd_kota from resume where
                            jum_mhs < @range_awal))
                            then @nilai1 * @bobot
                        when(kd_kota in (select kd_kota from resume where
                            jum_mhs >=@range_awal and jum_mhs <=@range_akhir))
                            then @nilai2 * @bobot
                        else @nilai3 * @bobot end";
    cmd = new SqlCommand(strSql, conn);
    cmd.Parameters.AddWithValue("@range_awal", rangeawal);
    cmd.Parameters.AddWithValue("@range_akhir", rangeakhir);
    cmd.Parameters.AddWithValue("@nilai1", nilai1);
    cmd.Parameters.AddWithValue("@nilai2", nilai2);
    cmd.Parameters.AddWithValue("@nilai3", nilai3);
    cmd.Parameters.AddWithValue("@bobot", bobotx);
    conn.Close();
    conn.Open();
    cmd.ExecuteNonQuery();
}
```

Kode Program 4.3 Cara Penilaian Data Jumlah Mahasiswa

Setelah data jumlah mahasiswa, sistem akan membandingkan jarak kota tujuan yang dimasukkan oleh pengguna dengan data di *database*, kemudian sistem akan memberi nilai pada setiap data jarak kota seperti yang terlihat pada Kode Program 4.4. Pemberian nilai dilanjutkan dengan mengalikan nilai dengan bobot kriteria jarak kota tujuan.


```
private void JarakKota()
{
    double bobotx = GetBobot(3);
    int nilai1 = 0; int nilai2 = 0; int nilai3 = 0;
    if (cmbJarak.SelectedItem.ToString() == "Lokal")
    { nilai1 = 100; nilai2 = 60; nilai3 = 30; }
    if (cmbJarak.SelectedItem.ToString() == "Dekat")
    { nilai1 = 60; nilai2 = 100; nilai3 = 30; }
    if (cmbJarak.SelectedItem.ToString() == "Jauh")
    { nilai1 = 30; nilai2 = 60; nilai3 = 100;}
    string strSql = @"update hasil set jarak=
                case when(kd_kota in (select kd_kota from resume where jarak = 1))
                        then @nilai1 * @bobot
                    when(kd_kota in (select kd_kota from resume where jarak = 2))
                        then @nilai2 * @bobot
                    else @nilai3 * @bobot end";
    cmd = new SqlCommand(strSql, conn);
    cmd.Parameters.AddWithValue("@nilai1", nilai1);
    cmd.Parameters.AddWithValue("@nilai2", nilai2);
    cmd.Parameters.AddWithValue("@nilai3", nilai3);
    cmd.Parameters.AddWithValue("@bobot", bobotx);
    conn.Close();conn.Open();
    cmd.ExecuteNonQuery();
}
```

Kode Program 4.4 Cara Penilaian Data Jarak Kota Tujuan

Setelah data jarak kota tujuan, sistem akan membandingkan jumlah SMA yang dimasukkan oleh pengguna dengan data di *database*, kemudian sistem akan memberi nilai pada setiap data jumlah SMA seperti yang terlihat pada Kode Program 4.5. Pemberian nilai dilanjutkan dengan mengalikan nilai dengan bobot kriteria jumlah SMA.

```
private void JumlahSMA()
{
    double bobotx = GetBobot(4);
    int rangeawal = Convert.ToInt32(txtJumSmaAwal.Text.Trim());
    int rangeakhir = Convert.ToInt32(txtJumSmaAkhir.Text.Trim());
    int nilai1 = 30; int nilai2 = 100; int nilai3 = 60;
    string strSql = @"update hasil set jum_sma=
                    case when(kd_kota in (select kd_kota from resume where
                            jum_sma < @range_awal)) then @nilai1 * @bobot
                         when(kd_kota in (select kd_kota from resume where
                            jum_sma >=@range_awal and jum_sma <=@range_akhir))
                            then @nilai2 * @bobot
                         else @nilai3 * @bobot end";
    cmd = new SqlCommand(strSql, conn);
    cmd.Parameters.AddWithValue("@range_awal", rangeawal);
    cmd.Parameters.AddWithValue("@range_akhir", rangeakhir);
    cmd.Parameters.AddWithValue("@nilai1", nilai1);
    cmd.Parameters.AddWithValue("@nilai2", nilai2);
    cmd.Parameters.AddWithValue("@nilai3", nilai3);
    cmd.Parameters.AddWithValue("@bobot", bobotx);
    conn.Close();
    conn.Open();
    cmd.ExecuteNonQuery();
}
```

Kode Program 4.5 Cara Penilaian Data Jumlah SMA

Setelah data jumlah SMA, sistem akan membandingkan rata-rata IPK yang dimasukkan oleh pengguna dengan data di *database*, kemudian sistem akan memberi nilai pada setiap data rata-rata IPK seperti yang terlihat pada Kode Program 4.6. Pemberian nilai dilanjutkan dengan mengalikan nilai dengan bobot kriteria jarak rata-rata IPK.

```
private void RerataIPK()
{
    double bobotx = GetBobot(5);
    int nilai1 = 0; int nilai2 = 0; int nilai3 = 0;
    if (cmbRrIpk.SelectedItem.ToString().Trim() == "≥ 0  sampai  < 2.05")
    { nilai1 = 100; nilai2 = 60; nilai3 = 30; }
    if (cmbRrIpk.SelectedItem.ToString().Trim() == "≥ 2.05  sampai  < 3")
    { nilai1 = 60; nilai2 = 100; nilai3 = 30; }
    if (cmbRrIpk.SelectedItem.ToString().Trim() == "≥ 3  sampai  ≤ 4")
    { nilai1 = 30; nilai2 = 60; nilai3 = 100; }
    string strSql = @"update hasil set rr_ipk=
                    case when(kd_kota in (select kd_kota from resume where rr_ipk >= 0
                       and rr_ipk < 2.05)) then @nilai1 * @bobot
                         when(kd_kota in (select kd_kota from resume where rr_ipk >= 2.05
                       and rr_ipk < 3)) then @nilai2 * @bobot
                         else @nilai3 * @bobot end";
    cmd = new SqlCommand(strSql, conn);
    cmd.Parameters.AddWithValue("@nilai1", nilai1);
    cmd.Parameters.AddWithValue("@nilai2", nilai2);
    cmd.Parameters.AddWithValue("@nilai3", nilai3);
    cmd.Parameters.AddWithValue("@bobot", bobotx);
    conn.Close();
    conn.Open();
    cmd.ExecuteNonQuery();
}
```

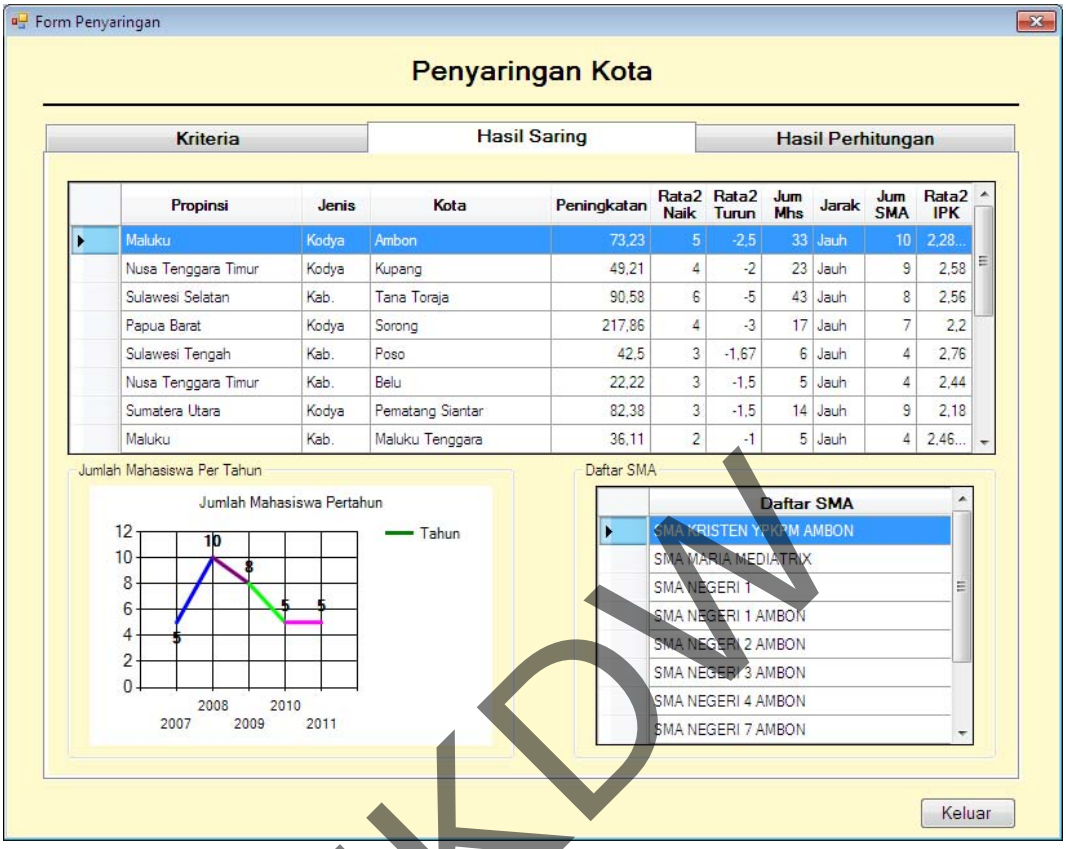
Kode Program 4.6 Cara Penilaian Data Rata-rata IPK

Setelah semua data kriteria diberi nilai dan dikalikan dengan bobot dari data kriteria tersebut, sistem akan menjumlahkannya berdasarkan kota seperti pada Kode Program 4.7.

```
private void JumlahNilai()
{
    int jum; string kd_kota;
    for (int x = 0; x < lstHasil.Count; x++)
    {
        kd_kota = lstHasil[x].kd_kota;
        jum = lstHasil[x].peningkatan + lstHasil[x].jum_mhs + lstHasil[x].jarak
              + lstHasil[x].jum_sma + lstHasil[x].rr_ipk;
        string strSql = @"update hasil set jumlah=@jumlah where kd_kota=@kd_kota";
        cmd = new SqlCommand(strSql, conn);
        cmd.Parameters.AddWithValue("@kd_kota", kd_kota);
        cmd.Parameters.AddWithValue("@jumlah", jum);
        conn.Close();
        conn.Open();
        cmd.ExecuteNonQuery();
    }
}
```

Kode Program 4.7 Cara Menjumlahkan Seluruh Nilai

Setelah semua perhitungan dilakukan, sistem akan menampilkan hasil penyaringan seperti pada Gambar 4.11. Kota yang hasil perhitungannya paling tinggi akan berada pada urutan pertama. Sistem akan menampilkan 25 kota dengan nilai tertinggi.



Gambar 4.11 Form Penyaringan Kota (Hasil Saring)

Untuk melihat hasil perhitungan, maka pengguna dapat memlih *tab menu* hasil perhitungan. Hasil perhitungan akan menunjukan nilai angka dari setiap kota berdasarkan kriteria-kriteria seperti terlihat pada gambar 4.12.

Gambar 4.12 Form Penyaringan Kota (Hasil Perhitungan)

4.2 Analisis Sistem

Pada analisis ini akan dilakukan tiga kali pengujian dengan bobot kriteria yang berubah-ubah namun masukan untuk setiap kriterianya selalu sama seperti pada Tabel 4.1.

Tabel 4.1 Masukan Kriteria

Kriteria	Masukan
Skala Rata-rata Peningkatan	75% – 200%
Skala Jumlah Mahasiswa	10 – 20
Jarak Kota Tujuan	Jauh
Skala Jumlah SMA	5 – 10
Rata-rata IPK	≥ 2.05 sampai ≤ 3
Perbandingan Rata-rata	Rata-rata Naik > Rata-rata Turun
Kota dengan data < 5 Tahun	Termasuk

a. Pengujian pertama

Pada pengujian pertama, bobot kriteria yang dimasukkan adalah sebagai berikut :

1) Peningkatan : 40
2) Jumlah mahasiswa : 20
3) Jarak kota tujuan : 20
4) Jumlah SMA : 15
5) Rata-rata IPK : 5

Berdasarkan bobot di atas, penyaringan akan menghasilkan kota-kota seperti gambar 4.13.

Propinsi	Jenis	Kota	Peningkatan	Rata2 Naik	Rata2 Turun	Jum Mhs	Jarak	Jum SMA	Rata2 IPK
Kalimantan Utara	Kodya	Tarakan	115,53	3	-2,5	12	Jauh	8	2,96
Papua	Kab.	Merauke	144,58	4	-3	12	Jauh	6	2,05
Sumatera Utara	Kodya	Pematang Siantar	82,38	3	-1,5	14	Jauh	9	2,18
Sulawesi Selatan	Kab.	Tana Toraja	90,58	6	-5	43	Jauh	8	2,56
Kalimantan Tengah	Kodya	Palangka Raya	158,33	3,5	-2	18	Jauh	13	1,87
Kalimantan Timur	Kodya	Bontang	100	1,67	-1	6	Jauh	5	1,42
Papua Barat	Kodya	Sorong	217,86	4	-3	17	Jauh	7	2,2
Bali	Kab.	Tabanan	79,17	3	-2	7	Dekat	5	2,46

Gambar 4.13 Hasil Pengujian Pertama

b. Pengujian kedua

Pada pengujian pertama, bobot kriteria yang dimasukkan adalah sebagai berikut :

1) Peningkatan : 20
2) Jumlah mahasiswa : 40
3) Jarak kota tujuan : 15
4) Jumlah SMA : 20
5) Rata-rata IPK : 5

Berdasarkan bobot di atas, penyaringan akan menghasilkan kota-kota seperti gambar 4.14.

Propinsi	Jenis	Kota	Peningkatan	Rata2 Naik	Rata2 Turun	Jum Mhs	Jarak	Jum SMA	Rata2 IPK
Kalimantan Utara	Kodya	Tarakan	115,53	3	-2,5	12	Jauh	8	2,96
Papua	Kab.	Merauke	144,58	4	-3	12	Jauh	6	2,05
Sumatera Utara	Kodya	Pematang Siantar	82,38	3	-1,5	14	Jauh	9	2,18
Papua Barat	Kodya	Sorong	217,86	4	-3	17	Jauh	7	2,2
Kalimantan Tengah	Kodya	Palangka Raya	158,33	3,5	-2	18	Jauh	13	1,87
Sulawesi Selatan	Kab.	Tana Toraja	90,58	6	-5	43	Jauh	8	2,56
Kalimantan Barat	Kodya	Pontianak	207,49	6	-2	22	Jauh	8	3,07
Kalimantan Timur	Kodya	Bontang	100	1,67	-1	6	Jauh	5	1,42

Gambar 4.14 Hasil Pengujian Kedua

c. Pengujian ketiga

Pada pengujian pertama, bobot kriteria yang dimasukkan adalah sebagai berikut :

1) Peningkatan : 20
2) Jumlah mahasiswa : 20
3) Jarak kota tujuan : 40
4) Jumlah SMA : 10
5) Rata-rata IPK : 10

Berdasarkan bobot di atas, penyaringan akan menghasilkan kota-kota seperti gambar 4.15.

Propinsi	Jenis	Kota	Peningkatan	Rata2 Naik	Rata2 Turun	Jum Mhs	Jarak	Jum SMA	Rata2 IPK
Kalimantan Utara	Kodya	Tarakan	115,53	3	-2,5	12	Jauh	8	2,96
Papua	Kab.	Merauke	144,58	4	-3	12	Jauh	6	2,05
Sumatera Utara	Kodya	Pematang Siantar	82,38	3	-1,5	14	Jauh	9	2,18
Papua Barat	Kodya	Sorong	217,86	4	-3	17	Jauh	7	2,2
Sulawesi Selatan	Kab.	Tana Toraja	90,58	6	-5	43	Jauh	8	2,56
Kalimantan Tengah	Kodya	Palangka Raya	158,33	3,5	-2	18	Jauh	13	1,87
Kalimantan Timur	Kodya	Bontang	100	1,67	-1	6	Jauh	5	1,42
Kalimantan Barat	Kab.	Pontianak	100	1	0	2	Jauh	2	2,38

Gambar 4.15 Hasil Pengujian Ketiga

Berdasarkan ketiga pengujian di atas, dapat dilihat bahwa hasil penyaringan tidak menunjukan perbedaan yang mencolok. Kota-kota yang dihasilkan cenderung sama karena kriteria yang dimasukan juga sama walaupun pembobotan kriteria pada setiap pengujian berbeda. Pembobotan kriteria yang berbeda pada setiap pengujian mempengaruhi nilai pada data kriteria tersebut. Jika kriteria peningkatan memiliki bobot yang paling tinggi, maka data dengan

peningkatan yang sesuai dengan masukan pengguna juga akan bernilai paling tinggi diantara yang lainnya. Oleh sebab itu, jika pengguna hendak menyaring kota-kota dengan memprioritaskan salah satu kriteria, maka pengguna perlu mengatur bobot kriteria tersebut dengan nilai bobot yang paling tinggi. Sebaliknya, jika pengguna hendak menyaring kota-kota dengan kriteria yang berbeda-beda namun bobot yang digunakan selalu sama, maka pengguna dapat mengatur kriteria tersebut saat memberi masukan pada setiap kriteria.

d. Pengujian keempat

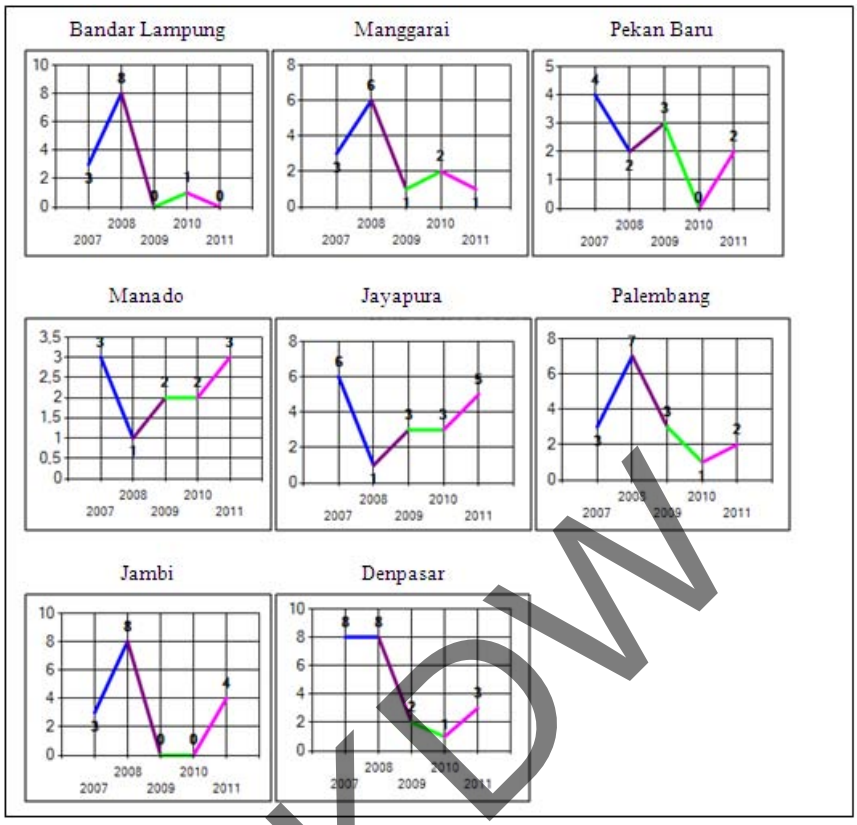
Pada pengujian keempat akan disaring kota-kota yang mengalami penurunan pada jumlah mahasiswa masuk setiap tahun menggunakan. Berikut ini adalah contoh masukan kriteria yang digunakan untuk menyaring kota-kota tersebut.

1) Skala rata-rata peningkatan : 30% - 120%
2) Skala jumlah mahasiswa : 10 - 50
3) Jarak kota tujuan : Jauh
4) Skala jumlah SMA : 3 - 10
5) Rata-rata IPK : ≥ 2.05 sampai ≤ 3
6) Perbandingan rata-rata : Rata-rata Naik < Rata-rata Turun
7) Kota dengan data < 5 tahun : Termasuk

Berdasarkan masukan di atas, penyaringan akan menghasilkan kota-kota seperti gambar 4.16 dan grafik jumlah mahasiswa masuk setiap tahun dari setiap kota seperti pada gambar 4.17.

Propinsi	Jenis	Kota	Peningkatan	Rata2 Naik	Rata2 Turun	Jum Mhs	Jarak	Jum SMA	Rata2 IPK
Lampung	Kodya	Bandar Lampung	68,94	3	-4,5	12	Jauh	9	2,37
Nusa Tenggara Timur	Kab.	Manggarai	59,86	2	-3	13	Jauh	7	2,32
Riau	Kodya	Pekanbaru	30,56	1,5	-2,5	11	Jauh	5	2,86
Sulawesi Utara	Kodya	Manado	38,54	1	-2	11	Jauh	5	2,66
Papua	Kab.	Jayapura	32	2	-5	18	Jauh	10	1,7
Sumatera Selatan	Kodya	Palembang	71,33	2,5	-3	16	Jauh	5	3,06
Jambi	Kodya	Jambi	75,76	4,5	-8	15	Jauh	9	3
Bali	Kodya	Denpasar	33,46	2	-3,5	22	Dekat	8	2,38

Gambar 4.16 Hasil Pengujian Keempat



Gambar 4.17 Grafik Jumlah Mahasiswa Masuk

Gambar 4.16 menunjukkan bahwa penyaringan menghasilkan empat kota yang berada pada urutan atassesuai dengan masukan kriteria, sedangkan kota-kota lainnya mendekati kriteria yang dimasukkan. Pada gambar 4.17 dapat dilihat bahwa grafik jumlah mahasiswa yang masuk dari setiap kota cukup beragam, tetapi secara perhitungan beberapa kota yang grafiknya terlihat naik pada tahun-tahun terakhir ternyata pernah mengalami jumlah penurunan yang cukup banyak dibanding dengan jumlah kenaikannya. Hal inilah yang menyebabkan kota-kota tersebut termasuk dalam penyaringan kota-kota yang mengalami penurunan.

Hasil penyaringan juga bergantung pada kesesuaian data. Jika data kota yang terdapat pada database sesuai dengan masukan kriteria yang diinginkan pengguna, maka sistem akan merekomendasikan kota-kota tersebut, namun jika tidak sesuai, sistem akan merekomendasikan kota-kota yang mendekati kriteria yang diinginkan pengguna.

4.3 Kelebihan dan Kekurangan Sistem

Dari hasil uji coba penggunaan sistem maka dapat dilihat kelebihan dan kekurangan sistem.

4.3.1 Kelebihan Sistem

a. Sistem mampu menghasilkan rekomendasi kota tujuan promosi berdasarkan kriteria-kriteria yang dimasukkan oleh pengguna.
b. Sistem mampu menentukan kota-kota tujuan promosi dengan memprioritaskan satu atau lebih kriteria berdasarkan bobot kriteria yang diatur oleh pengguna.
c. Sistem yang dibangun dengan menerapkan metode Multikriteria menentukan kota-kota yang sesuai dan mendekati keinginan pengguna.

4.3.2 Kekurangan Sistem

a. Sistem hanya menangani jenis kriteria yang ada dalam sistem saja, sehingga kriteria lain yang belum ada disistem tidak diperhitungkan.
b. Sistem tidak mampu mengolah pengetahuan dan pengalaman tim promosi dalam menentukan kota tujuan promosi.

LAMPIRAN

A. Listing Program

Form Login

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Data.SqlClient;
using System.Configuration;

namespace Skripsi0501
{
    public partial class Login : Form
    {
        private SqlConnection conn;
        private SqlCommand cmd;
        private SqlDataReader sdr;
        string hak_akses = "";

        public Login()
        {
            InitializeComponent();
        }

        private void btnBatal_Click(object sender, EventArgs e)
        {
            this.Close();
        }

        private Boolean CekMasuk(string username,string sandi)
        {
            string usernamex = "";
            Boolean cek = false;
            string strSql = @"select username, hak from users where username=@username and
            sandi=@sandi";
            cmd = new SqlCommand(strSql, conn);
            cmd.Parameters.AddWithValue("@username", username);
            cmd.Parameters.AddWithValue("@sandi", sandi);
            conn.Close();
            conn.Open();
            sdr = cmd.ExecuteReader();
            if (sdr.HasRows)
            {
                while (sdr.Read())
                {
                    usernamex = sdr["username"].ToString();
                    hak_akses = sdr["hak"].ToString();
                }
            }

            if (usernamex != "")
            {
                cek = true;
            }
            else
            {
                cek = false;
            }
            return cek;
```

```
        }

        private void btnMasuk_Click(object sender, EventArgs e)
        {
            if (CekMasuk(txtUsername.Text.Trim().ToString(),
            txtPass.Text.Trim().ToString()))
            {
                if (hak_akses == "Admin") //jika yang login sebagai admin
                {
                    MessageBox.Show("Anda login sebagai Admin", "Konfirmasi",
                    MessageBoxButtons.OK, MessageBoxIcon.Information,
                    MessageBoxDefaultButton.Button1);
                    MenuAdmin menuAdmin = new MenuAdmin();
                    if (menuAdmin.ShowDialog(this) == DialogResult.OK)
                    { }
                    else
                    {
                        menuAdmin.Dispose();
                        this.Activate();
                        this.Show();
                    }
                }
                else //jika yang login pengguna
                {
                    MessageBox.Show("Anda login sebagai Pengguna", "Konfirmasi",
                    MessageBoxButtons.OK, MessageBoxIcon.Information,
                    MessageBoxDefaultButton.Button1);
                    MenuKaryawan menuKaryawan = new MenuKaryawan();
                    if (menuKaryawan.ShowDialog(this) == DialogResult.OK)
                    { }
                    else
                    {
                        menuKaryawan.Dispose();
                        this.Activate();
                        this.Show();
                    }
                }
            }
            else
            {
                MessageBox.Show("Nama Pengguna / Kata Sandi tidak cocok", "Perhatian",
                MessageBoxButtons.OK, MessageBoxIcon.Stop,
                MessageBoxDefaultButton.Button1);
            }
        }

        private void Login_Load(object sender, EventArgs e)
        {
            string connStr = @"Data Source=.\SQLEXPRESS;Integrated Security=SSPI;Initial
            Catalog=Db_Skripsi";
            conn = new SqlConnection(connStr);
            conn.Open();
        }
    }
}
```


Form Menu Admin

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
```

```
using System.Text;
using System.Windows.Forms;
using System.Data.SqlClient;
using System.Configuration;
using System.Diagnostics;
using System.IO;

namespace Skripsi0501
{
    public partial class MenuAdmin : Form
    {
        #region variable
        private SqlDataAdapter da;
        private SqlCommandBuilder cb; //men-generate perintah insert update delete
        private DataSet ds;
        private BindingSource bs;
        private SqlConnection conn;
        private SqlCommand cmd;
        private SqlDataReader dr;
        string nmx = "";
        string je = "";
        string kd_kota = "";
        AlertForm alert;
        List<Kota> lstKota = new List<Kota>();
        List<Peningkatan> lstPen = new List<Peningkatan>();
        List<Resume> lstResume = new List<Resume>();
        #endregion

        public MenuAdmin()
        {
            InitializeComponent();
        }

        #region tombol
        private void button1_Click(object sender, EventArgs e)
        {
            SetupPropinsiDetail setupProp = new SetupPropinsiDetail();
            setupProp.ShowDialog();
            setupProp.Dispose();
            this.Activate();
            this.Show();
        }

        private void button2_Click(object sender, EventArgs e)
        {
            SetupKotaDetail setupKota = new SetupKotaDetail();
            setupKota.ShowDialog();
            setupKota.Dispose();
            this.Activate();
            this.Show();
        }

        private void button3_Click(object sender, EventArgs e)
        {
            SetupMahasiswaDetail setupMhs = new SetupMahasiswaDetail();
            setupMhs.ShowDialog();
            setupMhs.Dispose();
            this.Activate();
            this.Show();
        }

        private void button4_Click(object sender, EventArgs e)
        {
            SetupJarak setupJarak = new SetupJarak();
            setupJarak.ShowDialog();
            setupJarak.Dispose();
```

```
        this.Activate();
        this.Show();
    }

    private void button5_Click(object sender, EventArgs e)
    {
        SetupBobot setupBobot = new SetupBobot();
        setupBobot.ShowDialog();
        setupBobot.Dispose();
        this.Activate();
        this.Show();
    }

    private void button6_Click(object sender, EventArgs e)
    {
        SetupUser setupUser = new SetupUser();
        setupUser.ShowDialog();
        setupUser.Dispose();
        this.Activate();
        this.Show();
    }

    private void button7_Click(object sender, EventArgs e)
    {
        PenyaringanKota penyaringanKota = new PenyaringanKota();
        penyaringanKota.ShowDialog();
        penyaringanKota.Dispose();
        this.Activate();
        this.Show();
    }

    private void button8_Click(object sender, EventArgs e)
    {
        this.Close();
    }

    private void button9_Click(object sender, EventArgs e)
    {
        backgroundWorker1.WorkerReportsProgress = true;
        backgroundWorker1.WorkerSupportsCancellation = true;

        if (backgroundWorker1.IsBusy != true)
        {
            // create a new instance of the alert form
            alert = new AlertForm();
            // event handler for the Cancel button in AlertForm
            alert.Canceled += new EventHandler<EventArgs>(buttonCancel_Click);
            //this.Opacity = Convert.ToInt32(0);
            alert.Show();
            backgroundWorker1.RunWorkerAsync();
            // Start the asynchronous operation.
        }
    }
    #endregion

    #region proses
    private void IsiLstKota()
    {
        string strSql = @"select distinct(kd_kota) from mahasiswa";
        cmd = new SqlCommand(strSql, conn);
        conn.Close();
        conn.Open();
        dr = cmd.ExecuteReader();
        if (dr.HasRows)
        {
            while (dr.Read())
```

```
        {
            lstKota.Add(new Kota()
            {
                Kd_Kota = dr["kd_kota"].ToString()
            });
        }
    }
    conn.Close();
}

private int CountNim(string kodekotax, string tahunx)
{
    //hitung nim berdasarkan angkatan atau tahun masuk
    string strSql = @"select count(nim) as 'jum' from mahasiswa where
    kd_kota=@kd_kota and substring(nim,3,2) = @tahun";
    SqlCommand cmd = new SqlCommand(strSql, conn);
    cmd.Parameters.AddWithValue("@kd_kota", kodekotax.Trim());
    cmd.Parameters.AddWithValue("@tahun", tahunx.Trim());
    conn.Close();
    conn.Open();
    dr = cmd.ExecuteReader();
    dr.Read();
    return Convert.ToInt32(dr["jum"]);
}

private void HapusPeningkatan()
{
    string strSql = @"delete from peningkatan";
    cmd = new SqlCommand(strSql, conn);
    conn.Close();
    conn.Open();
    cmd.ExecuteNonQuery();
}

private void IsiPeningkatan()
{
    IsiLstKota();
    string kodekotax = "";
    int tahun1 = 0;
    int tahun2 = 0;
    int tahun3 = 0;
    int tahun4 = 0;
    int tahun5 = 0;
    for (int x = 0; x < lstKota.Count; x++)
    {
        kodekotax = lstKota[x].Kd_Kota.ToString();
        tahun1 = CountNim(kodekotax, "07");
        tahun2 = CountNim(kodekotax, "08");
        tahun3 = CountNim(kodekotax, "09");
        tahun4 = CountNim(kodekotax, "10");
        tahun5 = CountNim(kodekotax, "11");
        string strSql = @"insert into peningkatan (kd_kota, tahun1, tahun2, tahun3,
        tahun4, tahun5) values(@kd_kota,@tahun1,@tahun2,@tahun3,@tahun4,@tahun5)";
        cmd = new SqlCommand(strSql, conn);
        cmd.Parameters.AddWithValue("@kd_kota", kodekotax);
        cmd.Parameters.AddWithValue("@tahun1", tahun1);
        cmd.Parameters.AddWithValue("@tahun2", tahun2);
        cmd.Parameters.AddWithValue("@tahun3", tahun3);
        cmd.Parameters.AddWithValue("@tahun4", tahun4);
        cmd.Parameters.AddWithValue("@tahun5", tahun5);

        conn.Close();
        conn.Open();
        cmd.ExecuteNonQuery();
    }
}
```

```
private void IsiLstPeningkatan()
{
    string strSql = @"select kd_kota, tahun1, tahun2, tahun3, tahun4, tahun5 from
    peningkatan order by kd_kota";
    cmd = new SqlCommand(strSql, conn);
    conn.Close();
    conn.Open();
    dr = cmd.ExecuteReader();
    if (dr.HasRows)
    {
        while (dr.Read())
        {
            lstPen.Add(new Peningkatan()
            {
                kd_kota = dr["kd_kota"].ToString(),
                tahun1 = Convert.ToInt32(dr["tahun1"]),
                tahun2 = Convert.ToInt32(dr["tahun2"]),
                tahun3 = Convert.ToInt32(dr["tahun3"]),
                tahun4 = Convert.ToInt32(dr["tahun4"]),
                tahun5 = Convert.ToInt32(dr["tahun5"]),
            });
        }
    }
}

private void PresentaseTahunPeningkatan()
{
    double x1, x2, x3, x4;
    double itungNol, jum1, jum2, jum3, jum4, jum5;
    double rata2;
    for (int x = 0; x < lstPen.Count; x++)
    {
        itungNol = 0;
        jum1 = lstPen[x].tahun1; jum2 = lstPen[x].tahun2 + jum1;
        jum3 = lstPen[x].tahun3 + jum2; jum4 = lstPen[x].tahun4 + jum3;
        jum5 = lstPen[x].tahun5 + jum4;
        x1 = 0; x2 = 0; x3 = 0; x4 = 0; rata2 = 0;

        if (jum1 != 0)
            x1 = ((jum2 - jum1) / jum1) * 100;
        else if (jum1 == 0)
            itungNol++;

        if (jum2 != 0)
            x2 = ((jum3 - jum2) / jum2) * 100;
        else if (jum2 == 0)
            itungNol++;

        if (jum3 != 0)
            x3 = ((jum4 - jum3) / jum3) * 100;
        else if (jum3 == 0)
            itungNol++;

        if (jum4 != 0)
            x4 = ((jum5 - jum4) / jum4) * 100;
        else if (jum4 == 0)
            itungNol++;

        if (4 - itungNol == 0)
        {
            rata2 = 0;
        }
        else
        {
```

```
            rata2 = Convert.ToDouble(x1 + x2 + x3 + x4) / Convert.ToDouble(4 -
            itungNol);
        }

        string strSql = @"update peningkatan set jumlah=round(@jumlah,2) where
        kd_kota=@kd_kota";
        cmd = new SqlCommand(strSql, conn);
        cmd.Parameters.AddWithValue("@jumlah", Convert.ToDouble(rata2));
        cmd.Parameters.AddWithValue("@kd_kota", lstPen[x].kd_kota);
        conn.Close();
        conn.Open();
        cmd.ExecuteNonQuery();
    }

}

private void HapusHasil()
{
    string strSql = @"delete from hasil";
    cmd = new SqlCommand(strSql, conn);
    conn.Close();
    conn.Open();
    cmd.ExecuteNonQuery();
}

private void HapusResume()
{
    HapusHasil();
    string strSql = @"delete from resume";
    cmd = new SqlCommand(strSql, conn);
    conn.Close();
    conn.Open();
    cmd.ExecuteNonQuery();
}

private void IsiResume()
{
    string strSql = @"insert into resume(kd_kota, peningkatan, jum_mhs, jarak,
                    jum_sma, rr_ipk) select kota.kd_kota, peningkatan.jumlah,
                    count(nim), jarak, count(distinct(sma)), round((avg(ipk)),2)
                     from mahasiswa, kota, propinsi, peningkatan where
                    mahasiswa.kd_kota = kota.kd_kota and propinsi.kd_prop =
                    kota.kd_prop and kota.kd_kota = peningkatan.kd_kota
                     group by kota.kd_kota, peningkatan.jumlah, jarak";
    cmd = new SqlCommand(strSql, conn);
    conn.Close();
    conn.Open();
    cmd.ExecuteNonQuery();
}

private void ReratePeningkatan()
{
    double meningkat, menurun, rrmeningkat, rrmenurun;
    int itungtingkat, itungturun;
    for (int x = 0; x < lstPen.Count; x++)
    {
        meningkat = 0; menurun = 0; itungtingkat = 0; itungturun = 0; rrmeningkat =
        0; rrmenurun = 0;

        #region pertama
        if ((lstPen[x].tahun2 - lstPen[x].tahun1) > 0)
        {
            meningkat += lstPen[x].tahun2 - lstPen[x].tahun1;
            itungtingkat++;
        }
        else if ((lstPen[x].tahun2 - lstPen[x].tahun1) < 0)
```

```
{
    menurun += lstPen[x].tahun2 - lstPen[x].tahun1;
    itungturun++;
}
#endregion

#region kedua
if ((lstPen[x].tahun3 - lstPen[x].tahun2) > 0)
{
    meningkat += lstPen[x].tahun3 - lstPen[x].tahun2;
    itungtingkat++;
}
else if ((lstPen[x].tahun3 - lstPen[x].tahun2) < 0)
{
    menurun += lstPen[x].tahun3 - lstPen[x].tahun2;
    itungturun++;
}
#endregion

#region ketiga
if ((lstPen[x].tahun4 - lstPen[x].tahun3) > 0)
{
    meningkat += lstPen[x].tahun4 - lstPen[x].tahun3;
    itungtingkat++;
}
else if ((lstPen[x].tahun4 - lstPen[x].tahun3) < 0)
{
    menurun += lstPen[x].tahun4 - lstPen[x].tahun3;
    itungturun++;
}
#endregion

#region keempat
if ((lstPen[x].tahun5 - lstPen[x].tahun4) > 0)
{
    meningkat += lstPen[x].tahun5 - lstPen[x].tahun4;
    itungtingkat++;
}
else if ((lstPen[x].tahun5 - lstPen[x].tahun4) < 0)
{
    menurun += lstPen[x].tahun5 - lstPen[x].tahun4;
    itungturun++;
}
#endregion

if (itungtingkat != 0)
{
    rrmeningkat = meningkat / Convert.ToDouble(itungtingkat);
}
if (itungturun != 0)
{
    rrmenurun = menurun / Convert.ToDouble(itungturun);
}

string strSql = @"update peningkatan set
rr_meningkat=round(@rr_meningkat,2),rr_menurun=round(@rr_menurun,2) where
kd_kota=@kd_kota";
cmd = new SqlCommand(strSql, conn);
cmd.Parameters.AddWithValue("@rr_meningkat",
Convert.ToDouble(rrmeningkat));
cmd.Parameters.AddWithValue("@rr_menurun", Convert.ToDouble(rrmenurun));
cmd.Parameters.AddWithValue("@kd_kota",
lstPen[x].kd_kota.ToString().Trim());
conn.Close();
conn.Open();
cmd.ExecuteNonQuery();
```

```
    }
}
#endregion

#region bgworker
private void backgroundWorker1_DoWork(object sender, DoWorkEventArgs e)
{
    BackgroundWorker worker = sender as BackgroundWorker;

    for (int i = 1; i <= 10; i++)
    {
        if (worker.CancellationPending == true)
        {
            e.Cancel = true;
            break;
        }
        else
        {
            if (i == 1) // sebelum diisi, tabel peningkatan dihapus dulu
            { HapusPeningkatan();  }
            else if (i == 2) // isi tabel peningkatan
            { IsiPeningkatan(); }
            else if (i == 3) // isi list peningkatan berdasarkan tabel peningktan
            { IsiLstPeningkatan(); }
            else if (i == 4) // hitung presentase rata2 peningkatan lalu update
            tabel peningkatan
            { PresentaseTahunPeningkatan(); }
            else if (i == 5) // sebelum diisi, tabel resume dihapus dulu
            { HapusResume(); }
            else if (i == 6) // isi tabel resume
            { IsiResume(); }
            else // isi rata-rata naik dan rata-rata turun setiap kota
            { RerataPeningkatan(); }

            // Perform a time consuming operation and report progress.
            worker.ReportProgress(i * 10);
            System.Threading.Thread.Sleep(500);
        }
    }
}

private void backgroundWorker1_ProgressChanged(object sender,
 ProgressChangedEventArgs e)
{
    //Show the progress in main form (GUI)
    // Pass the progress to AlertForm label and progressbar
    alert.Message = "Sedang memproses, ... " + e.ProgressPercentage.ToString() +
    "%";
    alert.ProgressValue = e.ProgressPercentage;
}

private void backgroundWorker1_RunWorkerCompleted(object sender,
 RunWorkerCompletedEventArgs e)
{
    // Close the AlertForm
    alert.Close();
    ProsesResume prosesResume = new ProsesResume();
    prosesResume.ShowDialog();
    prosesResume.Dispose();
    this.Activate();
    this.Show();
}

private void buttonCancel_Click(object sender, EventArgs e)
{
    if (backgroundWorker1.WorkerSupportsCancellation == true)
```

```
            {
                // Cancel the asynchronous operation.
                backgroundWorker1.CancelAsync();
                // Close the AlertForm
                alert.Close();
            }
        }
        #endregion

        private void MenuAdmin_Load(object sender, EventArgs e)
        {
            string connStr = @"Data Source=.\SQLEXPRESS;Integrated Security=SSPI;Initial
           Catalog=Db_Skripsi";
            conn = new SqlConnection(connStr);
            conn.Open();
        }
    }
}
```

Form Menu Pengguna

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace Skripsi0501
{
    public partial class MenuKaryawan : Form
    {
        public MenuKaryawan()
        {
            InitializeComponent();
        }

        private void button7_Click(object sender, EventArgs e)
        {
            PenyaringanKota penyaringanKota = new PenyaringanKota();
            penyaringanKota.ShowDialog();
            penyaringanKota.Dispose();
            this.Activate();
            this.Show();
        }

        private void button8_Click(object sender, EventArgs e)
        {
            this.Close();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            UbahPassword ubahPass = new UbahPassword();
            ubahPass.ShowDialog();
            ubahPass.Dispose();
            this.Activate();
            this.Show();
        }

    }
}
```

Form Pengaturan Propinsi

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Data.SqlClient;
using System.Configuration;

namespace Skripsi0501
{
    public partial class SetupPropinsiDetail : Form
    {
        Boolean baru;
        string NamaPropx;
        string KodePropx;

        private SqlCommand cmd;
        private SqlConnection conn;
        private SqlDataReader dr;

        public SetupPropinsiDetail()
        {
            InitializeComponent();
        }

        private void SetupPropinsiDetail_Load(object sender, EventArgs e)
        {
            string connStr = @"Data Source=.\SQLEXPRESS;Integrated Security=SSPI;Initial
            Catalog=Db_Skripsi";
            conn = new SqlConnection(connStr);

            ModeLoad();
        }

        #region Mode
        private void ModeLoad()
        {
            txtNmProp.Enabled = false;
            cmbJarak.Enabled = false;

            btnNew.Enabled = true;
            btnEdit.Enabled = true;
            btnUndo.Enabled = false;
            btnSave.Enabled = false;
            btnDel.Enabled = true;
            btnDel.Visible = true;
            btnDel1.Visible = false;
            btnClose.Enabled = true;
        }

        private void ModeNew()
        {
            txtNmProp.Enabled = true;
            cmbJarak.Enabled = true;

            btnNew.Enabled = false;
            btnEdit.Enabled = false;
            btnUndo.Enabled = true;
            btnSave.Enabled = true;
            btnDel.Enabled = false;
```

```
    btnClose.Enabled = false;
}

private void ModeDel()
{
    txtNmProp.Enabled = false;
    cmbJarak.Enabled = false;

    btnNew.Enabled = false;
    btnEdit.Enabled = false;
    btnUndo.Enabled = true;
    btnSave.Enabled = false;
    btnDel.Visible = false;
    btnDel1.Visible = true;
    btnClose.Enabled = false;
}
#endregion

private void IsiDataSet()
{
    string strSql = @"select kd_prop, nm_prop, jarak from propinsi where
     kd_prop=@kd_prop";
    cmd = new SqlCommand(strSql, conn);
    cmd.Parameters.AddWithValue("@kd_prop", KodePropx);
    conn.Close();
    conn.Open();

    dr = cmd.ExecuteReader();

    if (dr.HasRows)
    {
        while (dr.Read())
        {
            txtNmProp.Text = dr["nm_prop"].ToString();
            cmbJarak.SelectedText = dr["jarak"].ToString();
            cmbJarak.SelectedItem = dr["jarak"].ToString();
        }
    }
}

private void GetKodeProp()
{
    string strSql = @"select kd_prop from propinsi where nm_prop=@nm_prop";
    cmd = new SqlCommand(strSql, conn);
    cmd.Parameters.AddWithValue("@nm_prop", NamaPropx);

    conn.Close();
    conn.Open();

    dr = cmd.ExecuteReader();
    if (dr.HasRows)
    {
        while (dr.Read())
        {
            KodePropx = (dr["kd_prop"]).ToString().Trim();
        }
    }
    conn.Close();
}

private string GetKodePropTerakhir()
{
    string kd_prop = "";
    int kd_prop_int = 0;
    string strSql = @"select top 1 kd_prop from propinsi order by
     SUBSTRING(kd_prop,2,2) desc";
```

```
            cmd = new SqlCommand(strSql, conn);
            conn.Close();
            conn.Open();
            dr = cmd.ExecuteReader();
            if (dr.HasRows)
            {
                while (dr.Read())
                {
                    kd_prop = dr["kd_prop"].ToString();
                }
            }
            kd_prop = kd_prop.Substring(1, 2); //ambil kode kota 3 digit terakhir
            kd_prop_int = Convert.ToInt32(kd_prop) + 1; // dijadikan integer trus tambahin
1
            kd_prop = "0000" + kd_prop_int.ToString(); // ditambah dengan string 0000
            return kd_prop = kd_prop.Substring(kd_prop.Length - 2, 2); // dipotong lagi
           ambil 3 digit paling belakang
            //conn.Close();
        }

        private void TambahProp()
        {
            string kodeterakhir = "P" + GetKodePropTerakhir();
            string strSql = @"insert into propinsi values(@kd_prop,@nm_prop,@jarak)";
            cmd = new SqlCommand(strSql, conn);
            cmd.Parameters.AddWithValue("@kd_prop", kodeterakhir);
            cmd.Parameters.AddWithValue("@nm_prop", txtNmProp.Text);
            cmd.Parameters.AddWithValue("@jarak", cmbJarak.SelectedItem.ToString().Trim());

            try
            {
                conn.Close();
                conn.Open();
                int status = cmd.ExecuteNonQuery();
                if (status == 1)
                {
                    MessageBox.Show("Data kota berhasil ditambahkan");
                }
            }
            catch (SqlException sqlEx)
            {
                if (sqlEx.Number == 2627) //nomor error klo primary key double
                    MessageBox.Show("Kode yang Anda inputkan sudah ada");
                else
                    MessageBox.Show("Error : " + sqlEx.Number.ToString() + " " +
                    sqlEx.Message);
            }
            finally
            {
                cmd.Dispose();
                conn.Close();
            }
        }

        private void UbahProp()
        {
            string strSql = @"update propinsi set nm_prop=@nm_prop, jarak=@jarak where
           kd_prop=@kd_prop";
            cmd = new SqlCommand(strSql, conn);
            cmd.Parameters.AddWithValue("kd_prop", KodePropx);
            cmd.Parameters.AddWithValue("nm_prop", txtNmProp.Text);
            cmd.Parameters.AddWithValue("@jarak", cmbJarak.SelectedItem.ToString().Trim());
            conn.Close();
            try
            {
                conn.Open();
```

```
            int status = cmd.ExecuteNonQuery();
            if (status == 1)
            {
                MessageBox.Show("Data kota berhasil diubah");
            }
        }
        catch (SqlException sqlEx)
        {
            MessageBox.Show("Error : " + sqlEx.Number.ToString() + " " +
            sqlEx.Message);
        }
        finally
        {
            cmd.Dispose();
            conn.Close();
        }
    }

    private void HapusProp()
    {
        string strSql = "delete from propinsi where kd_prop=@kd_prop";
        cmd = new SqlCommand(strSql, conn);
        cmd.Parameters.AddWithValue("@kd_prop", KodePropx);

        try
        {
            conn.Close();
            conn.Open();
            int result = cmd.ExecuteNonQuery();
            if (result == 1)
                MessageBox.Show("Data berhasil dihapus");
        }
        catch (SqlException sqlEx)
        {
            if (sqlEx.Number == 547)
            {
                MessageBox.Show("Data digunakan di tabel lain, tidak dapat dihapus.",
                "Perhatian", MessageBoxButtons.OK, MessageBoxIcon.Stop,
                MessageBoxDefaultButton.Button1);
            }
            else
            {
                MessageBox.Show("Error : " + sqlEx.Message);
            }
        }
        finally
        {
            cmd.Dispose();
            conn.Close();
        }
    }

    private void btnNew_Click(object sender, EventArgs e)
    {
        baru = true;
        ModeNew();
    }

    private void btnEdit_Click(object sender, EventArgs e)
    {
        baru = false;
        ModeNew();
        NamaPropx = "";
        CariPropinsi crProp = new CariPropinsi();
        if (crProp.ShowDialog(this) == DialogResult.OK)
        {
```

```
            this.txtNmProp.Text = crProp.NmProp;
            NamaPropx = crProp.NmProp;
        }
        crProp.Dispose();
        this.Activate();
        this.Show();

        if (!string.IsNullOrEmpty(NamaPropx))
        {
            GetKodeProp();
            IsiDataSet();
        }
        else
        {
            ModeLoad();
        }
    }

    private void btnDel1_Click(object sender, EventArgs e)
    {
        if (MessageBox.Show("Yakin menghapus?", "Konfirmasi", MessageBoxButtons.YesNo,
        MessageBoxIcon.Question, MessageBoxDefaultButton.Button2) ==
        System.Windows.Forms.DialogResult.Yes)
        {
            HapusProp();
            txtNmProp.Text = string.Empty;
            cmbJarak.Text = string.Empty;
            ModeLoad();
        }
    }

    private void btnDel_Click(object sender, EventArgs e)
    {
        ModeDel();
        NamaPropx = "";
        CariPropinsi crProp = new CariPropinsi();
        if (crProp.ShowDialog(this) == DialogResult.OK)
        {
            this.txtNmProp.Text = crProp.NmProp;
            NamaPropx = crProp.NmProp;
        }
        crProp.Dispose();
        this.Activate();
        this.Show();

        if (!string.IsNullOrEmpty(NamaPropx))
        {
            GetKodeProp();
            IsiDataSet();
        }
        else
        {
            ModeLoad();
        }
    }

    private void btnSave_Click(object sender, EventArgs e)
    {
        string a = txtNmProp.Text;
        var b = cmbJarak.SelectedItem;

        if (a == "" || b == null)
        {
            MessageBox.Show("Lengkapi semua data", "Perhatian", MessageBoxButtons.OK,
            MessageBoxIcon.Stop, MessageBoxDefaultButton.Button1);
        }
```

```
            else
            {
                if (baru)
                {
                    TambahProp();
                }
                else
                {
                    UbahProp();
                }
                txtNmProp.Text = string.Empty;
                cmbJarak.Text = string.Empty;
                ModeLoad();
            }
        }

        private void btnUndo_Click(object sender, EventArgs e)
        {
            txtNmProp.Text = string.Empty;
            cmbJarak.Text = string.Empty;
            ModeLoad();
        }

        private void btnClose_Click(object sender, EventArgs e)
        {
            this.Close();
        }
    }
}
```

Form Pencarian Propinsi

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Data.SqlClient;
using System.Configuration;

namespace Skripsi0501
{
    public partial class CariPropinsi : Form
    {
        private SqlCommand cmd;
        private SqlConnection conn;
        private BindingSource bs;
        private SqlDataAdapter da;
        private SqlCommandBuilder cb;
        private DataSet ds;

        public int indexBs;
        Boolean cek = false;

        public string NmProp { get; set; }

        public CariPropinsi()
        {
            InitializeComponent();
        }
```

```
        private void CariPropinsi_Load(object sender, EventArgs e)
        {
                        string connStr = @"Data Source=.\SQLEXPRESS;Integrated
            Security=SSPI;Initial Catalog=Db_Skripsi";
            conn = new SqlConnection(connStr);
        }

        private void cari()
        {
                        string strSql = @"select nm_prop, jarak from propinsi where nm_prop
            like '%' + @nm_prop + '%' order by nm_prop";
            cmd = new SqlCommand(strSql, conn);
            cmd.Parameters.AddWithValue("@nm_prop", txtCari.Text.Trim());

            conn.Close();
            conn.Open();

            da = new SqlDataAdapter();
            cb = new SqlCommandBuilder(da);
            da.SelectCommand = cmd;
            ds = new DataSet();
            da.Fill(ds, "Propinsi");
            bs = new BindingSource();
            bs.DataSource = ds.Tables["Propinsi"];
            dgvProp.DataSource = bs;
            dgvProp.Columns[0].HeaderCell.Value = "Propinsi";
            dgvProp.Columns[1].HeaderCell.Value = "Jarak";
            dgvProp.Columns[0].Width = 280;
            dgvProp.Columns[1].Width = 130;
        }

        private void btnCari_Click(object sender, EventArgs e)
        {
            cek = true;
            cari();
        }

        private void dgvProp_CellDoubleClick(object sender, DataGridViewCellEventArgs e)
        {
            if (cek)
            {
                DataRow dro = ds.Tables["Propinsi"].Rows[indexBs];
                NmProp = dro[0].ToString();
            }
            this.DialogResult = DialogResult.OK;
            this.Close();
        }

        private void dgvProp_CellClick(object sender, DataGridViewCellEventArgs e)
        {
            indexBs = bs.Position;
        }
    }
}
```

Form Pengaturan Kota

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
```

```
using System.Windows.Forms;
using System.Data.SqlClient;
using System.Configuration;

namespace Skripsi0501
{
    public partial class SetupKotaDetail : Form
    {
        Boolean baru;
        string NamaKotax;
        string JenisKotax;
        string KodeKotax;
        bool cekundo = false;
        private SqlCommand cmd;
        private SqlConnection conn;
        private SqlDataReader dr;

        public SetupKotaDetail()
        {
            InitializeComponent();
        }

        private void SetupKotaDetail_Load(object sender, EventArgs e)
        {
            string connStr = @"Data Source=.\SQLEXPRESS;Integrated Security=SSPI;Initial
           Catalog=Db_Skripsi";
            conn = new SqlConnection(connStr);

            ModeLoad();
        }

        #region Mode
        private void ModeLoad()
        {
            HapusListProp();
            cmbJenis.Enabled = false;
            txtNamaKota.Enabled = false;
            lstPropinsi.Enabled = false;

            btnNew.Enabled = true;
            btnEdit.Enabled = true;
            btnUndo.Enabled = false;
            btnSave.Enabled = false;
            btnDel.Enabled = true;
            btnDel.Visible = true;
            btnDel1.Visible = false;
            btnClose.Enabled = true;
        }

        private void ModeNew()
        {
            cekundo = false;
            cmbJenis.Enabled = true;
            txtNamaKota.Enabled = true;
            lstPropinsi.Enabled = true;

            btnNew.Enabled = false;
            btnEdit.Enabled = false;
            btnUndo.Enabled = true;
            btnSave.Enabled = true;
            btnDel.Enabled = false;
            btnClose.Enabled = false;
        }

        private void ModeDel()
        {
```

```
        cmbJenis.Enabled = false;
        txtNamaKota.Enabled = false;
        lstPropinsi.Enabled = false;

        btnNew.Enabled = false;
        btnEdit.Enabled = false;
        btnUndo.Enabled = true;
        btnSave.Enabled = false;
        btnDel.Visible = false;
        btnDel1.Visible = true;
        btnClose.Enabled = false;
    }
    #endregion

    private void IsiDataSet()
    {
        string strSql = @"select kd_kota,nm_kota,jenis,nm_prop from kota, propinsi
        where kota.kd_prop = propinsi.kd_prop and kd_kota=@kd_kota";
        cmd = new SqlCommand(strSql, conn);
        cmd.Parameters.AddWithValue("@kd_kota", KodeKotax);
        conn.Close();
        conn.Open();

        dr = cmd.ExecuteReader();

        if (dr.HasRows)
        {
            while (dr.Read())
            {
                lstPropinsi.SelectedItem = (dr["nm_prop"].ToString());
                txtPropinsi.Text = lstPropinsi.SelectedItem.ToString();
                txtNamaKota.Text = dr["nm_kota"].ToString();
                cmbJenis.Text = dr["jenis"].ToString();
            }
        }
    }

    private void GetAllPropinsi()
    {
        string strSql = @"select kd_prop, nm_prop from propinsi order by nm_prop asc";
        cmd = new SqlCommand(strSql, conn);
        conn.Close();
        conn.Open();
        int idx = 0;
        dr = cmd.ExecuteReader();

        if (dr.HasRows)
        {
            while (dr.Read())
            {
                lstPropinsi.Items.Add(dr["nm_prop"].ToString().Trim());
            }
        }
    }

    private void HapusListProp()
    {
        if (lstPropinsi.Items.Count > 0)
        {
            for (int a = lstPropinsi.Items.Count - 1; a >= 0; a--)
            {
                lstPropinsi.Items.RemoveAt(a);
            }
            lstPropinsi.Refresh();
        }
    }
```

```
private void GetKodeKota()
{
    string strSql = @"select kd_kota from kota where nm_kota=@nm_kota and
    jenis=@jenis";
    cmd = new SqlCommand(strSql, conn);
    cmd.Parameters.AddWithValue("@nm_kota", NamaKotax);
    cmd.Parameters.AddWithValue("@jenis", JenisKotax);

    conn.Close();
    conn.Open();

    dr = cmd.ExecuteReader();
    if (dr.HasRows)
    {
        while (dr.Read())
        {
            KodeKotax = (dr["kd_kota"]).ToString().Trim();
        }
    }
    conn.Close();
}

private string GetKodeProp()
{
    string a = lstPropinsi.SelectedItem.ToString().Trim();
    string strSql = @"select kd_prop from propinsi where nm_prop = @nm_prop";
    cmd = new SqlCommand(strSql, conn);
    cmd.Parameters.AddWithValue("@nm_prop", a);
    conn.Close();
    conn.Open();
    string b = "";
    dr = cmd.ExecuteReader();

    if (dr.HasRows)
    {
        while (dr.Read())
        {
            b = (dr["kd_prop"]).ToString();
        }
    }
    return b;
}

private string GetKodeKotaTerakhir()
{
    string kd_kota = "";
    int kd_kota_int = 0;
    string strSql = @"select top 1 kd_kota from kota order by
    SUBSTRING(Kd_Kota,4,3) desc";
    cmd = new SqlCommand(strSql, conn);
    conn.Close();
    conn.Open();
    dr.Close();
    dr = cmd.ExecuteReader();
    if (dr.HasRows)
    {
        while (dr.Read())
        {
            kd_kota = dr["kd_kota"].ToString();
        }
    }
    kd_kota = kd_kota.Substring(3, 3); //ambil kode kota 3 digit terakhir
    kd_kota_int = Convert.ToInt32(kd_kota) + 1; // dijadikan integer trus tambahin
1

    kd_kota = "0000" + kd_kota_int.ToString(); // ditambah dengan string 0000
```

```
    return kd_kota = kd_kota.Substring(kd_kota.Length - 3, 3); // dipotong lagi
    ambil 3 digit paling belakang
}

private void TambahKota()
{
    string kodepropx = GetKodeProp();
    string kodekotaterakhir ="K" + kodepropx.Substring(1,2) +
    GetKodeKotaTerakhir();
    string strSql = @"insert into kota values(@kd_kota, @nm_kota, @jenis,
    @kd_prop)";
    cmd = new SqlCommand(strSql, conn);
    cmd.Parameters.AddWithValue("@kd_kota", kodekotaterakhir.Trim().ToString());
    cmd.Parameters.AddWithValue("@nm_kota", txtNamaKota.Text.Trim());
    cmd.Parameters.AddWithValue("@jenis", cmbJenis.SelectedItem.ToString().Trim());
    cmd.Parameters.AddWithValue("@kd_prop", kodepropx.ToString().Trim());

    try
    {
        conn.Close();
        conn.Open();
        int status = cmd.ExecuteNonQuery();
        if (status == 1)
        {
            MessageBox.Show("Data Kota berhasil ditambah", "Konfirmasi",
            MessageBoxButtons.OK, MessageBoxIcon.Information);
        }
    }
    catch (SqlException sqlEx)
    {
        if (sqlEx.Number == 2627) //nomor error klo primary key double
            MessageBox.Show("Kode yang dimasukkan sudah ada", "Perhatian",
            MessageBoxButtons.OK, MessageBoxIcon.Stop);
        else
            MessageBox.Show("Error : " + sqlEx.Number.ToString() + " " +
            sqlEx.Message);
    }
    finally
    {
        cmd.Dispose();
        conn.Close();
    }
}

private void UbahKota()
{
    string kodepropx = GetKodeProp();
    string strSql = @"update kota set nm_kota=@nm_kota, jenis=@jenis,
    kd_prop=@kd_prop where kd_kota=@kd_kota";
    cmd = new SqlCommand(strSql, conn);
    cmd.Parameters.AddWithValue("@kd_kota", KodeKotax);
    cmd.Parameters.AddWithValue("@nm_kota", txtNamaKota.Text.Trim());
    cmd.Parameters.AddWithValue("@jenis", cmbJenis.SelectedItem.ToString().Trim());
    cmd.Parameters.AddWithValue("@kd_prop", kodepropx.ToString().Trim());

    try
    {
        conn.Close();
        conn.Open();
        int status = cmd.ExecuteNonQuery();
        if (status == 1)
        {
            MessageBox.Show("Data Kota berhasil diubah", "Konfirmasi",
            MessageBoxButtons.OK, MessageBoxIcon.Information);
        }
    }
```

```
    catch (SqlException sqlEx)
    {
        if (sqlEx.Number == 2627) //nomor error klo primary key double
            MessageBox.Show("Kode yang dimasukkan sudah ada", "Perhatian",
            MessageBoxButtons.OK, MessageBoxIcon.Stop);
        else
            MessageBox.Show("Error : " + sqlEx.Number.ToString() + " " +
            sqlEx.Message);
    }
    finally
    {
        cmd.Dispose();
        conn.Close();
    }
}

private void HapusKota()
{
    string strSql = "delete from kota where kd_kota=@kd_kota";
    cmd = new SqlCommand(strSql, conn);
    cmd.Parameters.AddWithValue("@kd_kota", KodeKotax);

    try
    {
        conn.Close();
        conn.Open();
        int result = cmd.ExecuteNonQuery();
        if (result == 1)
            MessageBox.Show("Data berhasil dihapus");
    }
    catch (SqlException sqlEx)
    {
        if (sqlEx.Number == 547)
        {
            MessageBox.Show("Data digunakan di tabel lain, tidak dapat dihapus.",
            "Perhatian", MessageBoxButtons.OK, MessageBoxIcon.Stop,
            MessageBoxDefaultButton.Button1);
        }
        else
        {
            MessageBox.Show("Error : " + sqlEx.Number);
        }
    }
    finally
    {
        cmd.Dispose();
        conn.Close();
    }
}

private void btnNew_Click(object sender, EventArgs e)
{
    baru = true;
    ModeNew();
    GetAllPropinsi();
}

private void btnUndo_Click(object sender, EventArgs e)
{
    cekundo = true;
    cmbJenis.Text = string.Empty;
    txtNamaKota.Text = string.Empty;
    txtPropinsi.Text = string.Empty;
    ModeLoad();
}
```

```
private void btnEdit_Click(object sender, EventArgs e)
{
    baru = false;
    ModeNew();
    NamaKotax = "";
    JenisKotax = "";
    CariKota crKota = new CariKota();
    if (crKota.ShowDialog(this) == DialogResult.OK)
    {
        this.txtNamaKota.Text = crKota.NmKota;
        NamaKotax = crKota.NmKota;
        JenisKotax = crKota.JnsKota;
    }
    crKota.Dispose();
    this.Activate();
    this.Show();

    if (!string.IsNullOrEmpty(NamaKotax))
    {
        GetAllPropinsi();
        GetKodeKota();
        IsiDataSet();
    }
    else
    {
        ModeLoad();
    }
}

private void btnClose_Click(object sender, EventArgs e)
{
    this.Close();
}

private void btnDel1_Click(object sender, EventArgs e)
{
    cekundo = true;
    if (MessageBox.Show("Yakin menghapus?", "Konfirmasi", MessageBoxButtons.YesNo,
    MessageBoxIcon.Question, MessageBoxDefaultButton.Button2) ==
    System.Windows.Forms.DialogResult.Yes)
    {
        HapusKota();
        cmbJenis.Text = string.Empty;
        txtNamaKota.Text = string.Empty;
        txtPropinsi.Text = string.Empty;
        ModeLoad();
    }
}

private void btnDel_Click(object sender, EventArgs e)
{
    ModeDel();
    NamaKotax = "";
    JenisKotax = "";
    CariKota crKota = new CariKota();
    if (crKota.ShowDialog(this) == DialogResult.OK)
    {
        this.txtNamaKota.Text = crKota.NmKota;
        NamaKotax = crKota.NmKota;
        JenisKotax = crKota.JnsKota;
    }
    crKota.Dispose();
    this.Activate();
    this.Show();

    if (!string.IsNullOrEmpty(NamaKotax))
```

```
            {
                GetAllPropinsi();
                GetKodeKota();
                IsiDataSet();
            }
            else
            {
                ModeLoad();
            }
        }

        private void btnSave_Click(object sender, EventArgs e)
        {
            cekundo = true;
            string a = lstPropinsi.SelectedItem.ToString();
            var b = cmbJenis.SelectedItem;
            string c = txtNamaKota.Text;

            if (a == "" || b == null || c == "")
            {
                MessageBox.Show("Lengkapi semua data", "Perhatian", MessageBoxButtons.OK,
                MessageBoxIcon.Stop, MessageBoxDefaultButton.Button1);
            }
            else
            {
                if (baru)
                {
                    TambahKota();
                }
                else
                {
                    UbahKota();
                }
                cmbJenis.Text = string.Empty;
                txtNamaKota.Text = string.Empty;
                txtPropinsi.Text = string.Empty;
                ModeLoad();
            }
        }

        private void lstPropinsi_SelectedIndexChanged(object sender, EventArgs e)
        {
            if (!cekundo)
            {
                txtPropinsi.Text = lstPropinsi.SelectedItem.ToString();
            }
        }
    }
}
```

Form Pencarian Kota

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Data.SqlClient;
using System.Configuration;

namespace Skripsi0501
```

```
{
    public partial class CariKota : Form
    {
        private SqlCommand cmd;
        private SqlConnection conn;
        private BindingSource bs;
        private SqlDataAdapter da;
        private SqlCommandBuilder cb;
        private DataSet ds;

        public int indexBs;
        Boolean cek = false;

        public string NmKota { get; set; }
        public string JnsKota { get; set; }

        public CariKota()
        {
            InitializeComponent();
        }

        private void cari()
        {
            if (cmbCari.SelectedItem.ToString() == "Kota")
            {
                string strSql = @"select jenis, nm_kota, nm_prop from kota, propinsi where
                kota.kd_prop = propinsi.kd_prop and nm_kota like '%' + @nm_kota + '%' order
                by nm_kota";
                cmd = new SqlCommand(strSql, conn);
                cmd.Parameters.AddWithValue("@nm_kota", txtCari.Text.Trim());
            }
            else
            {
                string strSql = @"select jenis, nm_kota, nm_prop from kota, propinsi where
                kota.kd_prop = propinsi.kd_prop and nm_prop like '%' + @nm_prop + '%' order
                by nm_kota";
                cmd = new SqlCommand(strSql, conn);
                cmd.Parameters.AddWithValue("@nm_prop", txtCari.Text.Trim());
            }

            conn.Close();
            conn.Open();

            da = new SqlDataAdapter();
            cb = new SqlCommandBuilder(da);
            da.SelectCommand = cmd;
            ds = new DataSet();
            da.Fill(ds, "kota");
            bs = new BindingSource();
            bs.DataSource = ds.Tables["kota"];
            dgvKota.DataSource = bs;

            //style dgv
            dgvKota.Columns[0].HeaderCell.Value = "Jenis";
            dgvKota.Columns[1].HeaderCell.Value = "Kota";
            dgvKota.Columns[2].HeaderCell.Value = "Propinsi";
            dgvKota.Columns[0].Width = 53;
            dgvKota.Columns[1].Width = 167;
            dgvKota.Columns[2].Width = 167;
        }

        private void btnCari_Click(object sender, EventArgs e)
        {
            cek = true;
            cari();
        }
```

```
        private void CariKota_Load(object sender, EventArgs e)
        {
            string connStr = @"Data Source=.\SQLEXPRESS;Integrated Security=SSPI;Initial
           Catalog=Db_Skripsi";
            conn = new SqlConnection(connStr);

            cmbCari.SelectedItem = "Kota";
        }

        private void btnPilih_Click(object sender, EventArgs e)
        {
            if (cek)
            {
                DataRow dro = ds.Tables["Kota"].Rows[indexBs];
                JnsKota = dro[0].ToString();
                NmKota = dro[1].ToString();
            }
            this.DialogResult = DialogResult.OK;
            this.Close();
        }

        private void dgvKota_CellDoubleClick(object sender, DataGridViewCellEventArgs e)
        {
            if (cek)
            {
                DataRow dro = ds.Tables["Kota"].Rows[indexBs];
                JnsKota = dro[0].ToString();
                NmKota = dro[1].ToString();
            }
            this.DialogResult = DialogResult.OK;
            this.Close();
        }

        private void dgvKota_CellClick(object sender, DataGridViewCellEventArgs e)
        {
            indexBs = bs.Position;
        }
    }
}
```


Form Pengaturan Mahasiswa

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using Microsoft.VisualBasic;
using System.Data.SqlClient;
using System.Configuration;

namespace Skripsi0501
{
    public partial class SetupMahasiswaDetail : Form
    {
        string cari_nim_mhs;
        Boolean baru;
        string NamaKotax;
        string JenisKotax;
        string KodeKotax;
```

```
string kodeKotaLama;
Boolean cek =false;

private SqlCommand cmd;
private SqlConnection conn;
private SqlDataAdapter da;
private SqlCommandBuilder cb;
private DataSet ds;
private BindingSource bs;
private SqlDataReader dr;
List<DaftarKota> lstDaftarKota = new List<DaftarKota>();
public SetupMahasiswaDetail()
{
    InitializeComponent();
}

private void SetupMahasiswaDetail_Load(object sender, EventArgs e)
{
    string connStr = @"Data Source=.\SQLEXPRESS;Integrated Security=SSPI;Initial
   Catalog=Db_Skripsi";
    conn = new SqlConnection(connStr);

    ModeLoad();
}

private void IsiDataSet()
{
    string strSql = @"select nim, sma, ipk, agama,mahasiswa.kd_kota, nm_kota, jenis
   from mahasiswa, kota where mahasiswa.kd_kota = kota.kd_kota and nim=@nim";
    cmd = new SqlCommand(strSql, conn);
    cmd.Parameters.AddWithValue("@nim", cari_nim_mhs);
    conn.Close();
    conn.Open();

    dr = cmd.ExecuteReader();

    if (dr.HasRows)
    {
        while (dr.Read())
        {
            if (dr["nim"] == null)
            {
                MessageBox.Show("NIM yang Anda cari tidak ditemukan", "Konfimasi",
                MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
            else
            {
                foreach (var lst in lstDaftarKota)
                {
                    if (lst.kd_kota == dr["kd_kota"].ToString())
                    {
                        lstKota.SetSelected(lst.no, true);
                    }
                }
                txtNim.Text = dr["nim"].ToString();
                txtKota.Text = dr["jenis"].ToString().Trim() + " "+
                dr["nm_kota"].ToString().Trim();
                txtSMA.Text = dr["sma"].ToString();
                txtIpk.Text = dr["ipk"].ToString();
                cmbAgama.Text = dr["agama"].ToString();
                kodeKotaLama = dr["kd_kota"].ToString();
            }
        }
    }
}
```

```
private void GetKodeKota()
{
    string strSql = @"select kd_kota from kota where nm_kota=@nm_kota and
    jenis=@jenis";
    cmd = new SqlCommand(strSql, conn);
    cmd.Parameters.AddWithValue("@nm_kota", NamaKotax);
    cmd.Parameters.AddWithValue("@jenis", JenisKotax);

    conn.Close();
    conn.Open();

    dr = cmd.ExecuteReader();
    if (dr.HasRows)
    {
        while (dr.Read())
        {
            KodeKotax = (dr["kd_kota"]).ToString().Trim();
        }
    }
    conn.Close();
}

private void GetAllKota()
{
    int counter = 0;
    string strSql = @"select kd_kota, jenis, nm_kota from kota order by nm_kota";
    cmd = new SqlCommand(strSql, conn);
    conn.Close();
    conn.Open();
    dr = cmd.ExecuteReader();
    if (dr.HasRows)
    {
        while (dr.Read())
        {
            lstKota.Items.Add(dr["nm_kota"].ToString().Trim() + " (" +
            dr["jenis"].ToString().Trim() + ")");
            lstDaftarKota.Add(new DaftarKota()
            {
                no = counter++,
                kd_kota = dr["kd_kota"].ToString().Trim(),
                jenis = dr["jenis"].ToString().Trim(),
                nm_kota = dr["nm_kota"].ToString().Trim()
            });
        }
    }

}

private void HapusListKota()
{
    if (lstKota.Items.Count > 0)
    {
        for (int a = lstKota.Items.Count - 1; a > -1; a--)
        {
            lstKota.Items.RemoveAt(a);
        }
        lstKota.Refresh();
    }
}

private void TambahMhs()
{
    string strSql = @"insert into mahasiswa(nim,sma,agama,kd_kota,ipk) values(@nim,
    @sma, @agama, @kd_kota, @ipk)";
    cmd = new SqlCommand(strSql, conn);
    cmd.Parameters.AddWithValue("@nim", txtNim.Text.Trim());
```

```
        cmd.Parameters.AddWithValue("@sma", txtSMA.Text.Trim().ToUpper());
        cmd.Parameters.AddWithValue("@ipk", Convert.ToDouble(txtIpk.Text.Trim()));
        cmd.Parameters.AddWithValue("@agama", cmbAgama.SelectedItem.ToString().Trim());
        cmd.Parameters.AddWithValue("@kd_kota", KodeKotax);

        try
        {
            conn.Close();
            conn.Open();
            int status = cmd.ExecuteNonQuery();
            if (status == 1)
            {
                MessageBox.Show("Data Mahasiswa berhasil
                ditambah","Konfirmasi",MessageBoxButtons.OK,MessageBoxIcon.Information)
                ;
            }
        }
        catch (SqlException sqlEx)
        {
            if (sqlEx.Number == 2627) //nomor error klo primary key double
                MessageBox.Show("NIM yang dimasukkan sudah ada","Perhatian",
                MessageBoxButtons.OK, MessageBoxIcon.Stop);
            else
                MessageBox.Show("Error : " + sqlEx.Number.ToString() + " " +
                sqlEx.Message);
        }
        finally
        {
            cmd.Dispose();
            conn.Close();
        }
    }

    private void UbahMhs()
    {
        string strSql = @"update mahasiswa set sma=@sma, ipk=@ipk, agama=@agama,
        kd_kota=@kd_kota where nim=@nim";
        cmd = new SqlCommand(strSql, conn);
        cmd.Parameters.AddWithValue("@nim", txtNim.Text.Trim());
        cmd.Parameters.AddWithValue("@sma", txtSMA.Text.Trim().ToUpper());
        cmd.Parameters.AddWithValue("@ipk", Convert.ToDouble(txtIpk.Text.Trim()));
        cmd.Parameters.AddWithValue("@agama", cmbAgama.SelectedItem.ToString().Trim());
        cmd.Parameters.AddWithValue("@kd_kota", KodeKotax);
        try
        {
            conn.Close();
            conn.Open();
            int status = cmd.ExecuteNonQuery();
            if (status == 1)
            {
                MessageBox.Show("Data Mahasiswa berhasil diubah", "Konfirmasi",
                MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
        }
        catch (SqlException sqlEx)
        {
            if (sqlEx.Number == 2627) //nomor error klo primary key double
                MessageBox.Show("NIM yang dimasukkan sudah ada", "Perhatian",
                MessageBoxButtons.OK, MessageBoxIcon.Stop);
            else
                MessageBox.Show("Error : " + sqlEx.Number.ToString() + " " +
                sqlEx.Message);
        }
        finally
        {
            cmd.Dispose();
```

```
            conn.Close();
        }
    }

    private void HapusMhs()
    {
        string strSql = "delete from mahasiswa where nim=@nim";
        cmd = new SqlCommand(strSql, conn);
        cmd.Parameters.AddWithValue("@nim", cari_nim_mhs);

        try
        {
            conn.Close();
            conn.Open();
            int result = cmd.ExecuteNonQuery();
            if (result == 1)
                MessageBox.Show("Data berhasil dihapus");
        }
        catch (SqlException sqlEx)
        {
            MessageBox.Show("Error : " + sqlEx.Message);
        }
        finally
        {
            cmd.Dispose();
            conn.Close();
        }
    }

    #region Mode
    private void ModeLoad()
    {
        HapusListKota();
        txtNim.Enabled = false;
        txtKota.Enabled = false;
        txtIpk.Enabled = false;
        txtSMA.Enabled = false;
        cmbAgama.Enabled = false;
        lstKota.Enabled = false;

        btnNew.Enabled = true;
        btnEdit.Enabled = true;
        btnUndo.Enabled = false;
        btnSave.Enabled = false;
        btnDel.Enabled = true;
        btnDel.Visible = true;
        btnDel1.Visible = false;
        btnClose.Enabled = true;

        cek = false;

    }

    private void ModeNew()
    {
        GetAllKota();
        txtNim.Enabled = true;
        txtKota.Enabled = false;
        txtIpk.Enabled = true;
        txtSMA.Enabled = true;
        cmbAgama.Enabled = true;
        lstKota.Enabled = true;

        btnNew.Enabled = false;
        btnEdit.Enabled = false;
        btnUndo.Enabled = true;
```

```
    btnSave.Enabled = true;
    btnDel.Enabled = false;
    btnClose.Enabled = false;
}

private void ModeDel()
{
    txtNim.Enabled = false;
    txtKota.Enabled = false;
    txtIpk.Enabled = false;
    txtSMA.Enabled = false;
    cmbAgama.Enabled = false;
    lstKota.Enabled = false;

    btnNew.Enabled = false;
    btnEdit.Enabled = false;
    btnUndo.Enabled = true;
    btnSave.Enabled = false;
    btnDel.Visible = false;
    btnDel1.Visible = true;
    btnClose.Enabled = false;
}
#endregion

private void btnClose_Click(object sender, EventArgs e)
{
    this.Close();
}

private void btnDel_Click(object sender, EventArgs e)
{
    ModeDel();
    cari_nim_mhs = "";
    CariMahasiswa crMhs = new CariMahasiswa();
    if (crMhs.ShowDialog(this) == DialogResult.OK)
    {
        cari_nim_mhs = crMhs.nim;
    }
    crMhs.Dispose();
    this.Activate();
    this.Show();

    if (!string.IsNullOrEmpty(cari_nim_mhs))
    {
        GetAllKota();
        IsiDataSet();
    }
    else
    {
        ModeLoad();
    }
}

private void btnEdit_Click(object sender, EventArgs e)
{
    baru = false;
    ModeNew();
    cari_nim_mhs = "";
    CariMahasiswa crMhs = new CariMahasiswa();
    if (crMhs.ShowDialog(this) == DialogResult.OK)
    {
        cari_nim_mhs = crMhs.nim;
    }
    crMhs.Dispose();
    this.Activate();
    this.Show();
```

```
    if (!string.IsNullOrEmpty(cari_nim_mhs))
    {
        GetAllKota();
        IsiDataSet();
        txtNim.Enabled = false;
        txtSMA.Focus();
    }
    else
    {
        ModeLoad();
    }

}

private void btnNew_Click(object sender, EventArgs e)
{
    baru = true;
    ModeNew();
    txtNim.Focus();
}

private void btnUndo_Click(object sender, EventArgs e)
{
    txtNim.Text = string.Empty;
    txtSMA.Text = string.Empty;
    txtKota.Text = string.Empty;
    txtIpk.Text = string.Empty;
    cmbAgama.Text = string.Empty;
    ModeLoad();
}

private void btnSave_Click(object sender, EventArgs e)
{
    string a = txtNim.Text;
    string b = txtKota.Text;
    string c = txtSMA.Text;
    var d = cmbAgama.SelectedItem;
    string f = txtIpk.Text;
    string aa = a.Substring(0, 2);

    if (a == "" || b == "" || c == "" || d == null || f == "")
    {
        MessageBox.Show("Lengkapi semua data", "Perhatian", MessageBoxButtons.OK,
        MessageBoxIcon.Stop, MessageBoxDefaultButton.Button1);
    }
    else if ((aa != "23" && aa != "21" && aa != "72" && aa != "71" && aa != "12" &&
    aa != "11")|| a.Length != 8)
    {
        MessageBox.Show("NIM yang dimasukkan salah", "Perhatian",
        MessageBoxButtons.OK, MessageBoxIcon.Stop,
        MessageBoxDefaultButton.Button1);
    }
    else if (Convert.ToDouble(f) > 4)
    {
        MessageBox.Show("Masukan IPK salah", "Perhatian", MessageBoxButtons.OK,
        MessageBoxIcon.Stop, MessageBoxDefaultButton.Button1);
    }
    else
    {
        if (baru)
        {
            TambahMhs();
        }
        else
        {
```

```
            UbahMhs();
        }
        txtNim.Text = string.Empty;
        txtSMA.Text = string.Empty;
        txtKota.Text = string.Empty;
        txtIpk.Text = string.Empty;
        cmbAgama.Text = string.Empty;
        ModeLoad();
    }

}

private void btnCariKota_Click(object sender, EventArgs e)
{
    cek = true;
    CariKota crKota = new CariKota();
    if (crKota.ShowDialog(this) == DialogResult.OK)
    {
        this.txtKota.Text = crKota.NmKota;
        NamaKotax = crKota.NmKota;
        JenisKotax = crKota.JnsKota;
    }
    crKota.Dispose();
    this.Activate();
    this.Show();
}

private void btnDel1_Click(object sender, EventArgs e)
{
    if (MessageBox.Show("Yakin menghapus?", "Konfirmasi", MessageBoxButtons.YesNo,
    MessageBoxIcon.Question, MessageBoxDefaultButton.Button2) ==
    System.Windows.Forms.DialogResult.Yes)
    {
        HapusMhs();
        txtNim.Text = string.Empty;
        txtSMA.Text = string.Empty;
        txtKota.Text = string.Empty;
        txtIpk.Text = string.Empty;
        cmbAgama.Text = string.Empty;
        ModeLoad();
    }
}

private void txtIpk_KeyPress(object sender, KeyPressEventArgs e)
{
    if (!char.IsControl(e.KeyChar) && !char.IsDigit(e.KeyChar) && e.KeyChar != '.')
    {
        e.Handled = true;
    }

    // only allow one decimal point
    if (e.KeyChar == '.' && (sender as TextBox).Text.IndexOf('.') > -1)
    {
        e.Handled = true;
    }
}

private void lstKota_SelectedIndexChanged(object sender, EventArgs e)
{
    foreach (var lst in lstDaftarKota)
    {
        if (lst.no == lstKota.SelectedIndex)
        {
            KodeKotax = lst.kd_kota;
            txtKota.Text = lst.jenis + " "+ lst.nm_kota;
        }
```

```
            }
        }
    }
}
```

Form Pencarian Mahasiswa

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Data.SqlClient;
using System.Configuration;

namespace Skripsi0501
{
    public partial class CariMahasiswa : Form
    {
        private SqlCommand cmd;
        private SqlConnection conn;
        private BindingSource bs;
        private SqlDataAdapter da;
        private SqlCommandBuilder cb;
        private DataSet ds;

        public int indexBs;
        Boolean cek = false;

        public string nim { get; set; }

        public CariMahasiswa()
        {
            InitializeComponent();
        }

        private void CariMahasiswa_Load(object sender, EventArgs e)
        {
            string connStr = @"Data Source=.\SQLEXPRESS;Integrated Security=SSPI;Initial
            Catalog=Db_Skripsi";
            conn = new SqlConnection(connStr);

            cmbCari.SelectedItem = "NIM";
        }

        private void cari()
        {
            if (cmbCari.SelectedItem.ToString() == "NIM")
            {
                string strSql = @"select nim, jenis, nm_kota, sma, ipk, agama from
                mahasiswa, kota where mahasiswa.kd_kota = kota.kd_kota and nim like '%' +
                @nim + '%' order by nim";
                cmd = new SqlCommand(strSql, conn);
                cmd.Parameters.AddWithValue("@nim", txtCari.Text.Trim());
            }
            else if (cmbCari.SelectedItem.ToString() == "Kota")
            {
                string strSql = @"select nim, jenis, nm_kota, sma, ipk, agama from
                mahasiswa, kota where mahasiswa.kd_kota = kota.kd_kota and nm_kota like '%'
                + @nm_kota + '%' order by nim";
                cmd = new SqlCommand(strSql, conn);
```

```
            cmd.Parameters.AddWithValue("@nm_kota", txtCari.Text.Trim());
        }
        else if (cmbCari.SelectedItem.ToString() == "SMA")
        {
            string strSql = @"select nim, jenis, nm_kota, sma, ipk, agama from
            mahasiswa, kota where mahasiswa.kd_kota = kota.kd_kota and sma like '%' +
            @sma + '%' order by nim";
            cmd = new SqlCommand(strSql, conn);
            cmd.Parameters.AddWithValue("@sma", txtCari.Text.Trim());
        }
        else if (cmbCari.SelectedItem.ToString() == "IPK")
        {
                            string strSql = @"select nim, jenis, nm_kota, sma, ipk,
            agama from mahasiswa, kota where mahasiswa.kd_kota = kota.kd_kota and IPK =
            @ipk order by nim";
            cmd = new SqlCommand(strSql, conn);
            cmd.Parameters.AddWithValue("@ipk", Convert.ToDouble(txtCari.Text.Trim()));
        }
        else
        {
            string strSql = @"select nim, jenis, nm_kota, sma, ipk, agama from
            mahasiswa, kota where mahasiswa.kd_kota = kota.kd_kota and agama like '%' +
            @agama + '%' order by nim";
            cmd = new SqlCommand(strSql, conn);
            cmd.Parameters.AddWithValue("@agama", txtCari.Text.Trim());
        }

        conn.Close();
        conn.Open();

        da = new SqlDataAdapter();
        cb = new SqlCommandBuilder(da);
        da.SelectCommand = cmd;
        ds = new DataSet();
        da.Fill(ds, "mahasiswa");
        bs = new BindingSource();
        bs.DataSource = ds.Tables["mahasiswa"];
        dgvMahasiswa.DataSource = bs;

        //style dgv
        dgvMahasiswa.Columns[0].HeaderCell.Value = "NIM";
        dgvMahasiswa.Columns[1].HeaderCell.Value = "Jenis";
        dgvMahasiswa.Columns[2].HeaderCell.Value = "Kota";
        dgvMahasiswa.Columns[3].HeaderCell.Value = "SMA";
        dgvMahasiswa.Columns[4].HeaderCell.Value = "IPK";
        dgvMahasiswa.Columns[5].HeaderCell.Value = "Agama";
        dgvMahasiswa.Columns[0].Width = 60;
        dgvMahasiswa.Columns[1].Width = 53;
        dgvMahasiswa.Columns[2].Width = 170;
        dgvMahasiswa.Columns[3].Width = 188;
        dgvMahasiswa.Columns[4].Width = 40;
        dgvMahasiswa.Columns[5].Width = 60;
    }

    private void btnCari_Click(object sender, EventArgs e)
    {
        cek = true;
        cari();
    }

    private void dgvMahasiswa_CellDoubleClick(object sender, DataGridViewCellEventArgs
     e)
    {
        if (cek)
        {
            DataRow dro = ds.Tables["Mahasiswa"].Rows[indexBs];
```

```
                nim = dro[0].ToString();
            }
            this.DialogResult = DialogResult.OK;
            this.Close();
        }

        private void dgvMahasiswa_CellClick(object sender, DataGridViewCellEventArgs e)
        {
            indexBs = bs.Position;
        }
    }
}
```

Form Pengaturan Bobot Kriteria

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Data.SqlClient;
using System.Configuration;

namespace Skripsi0501
{
    public partial class SetupBobot : Form
    {
        private SqlCommand cmd;
        private SqlConnection conn;
        private SqlDataReader dr;

        public SetupBobot()
        {
            InitializeComponent();
        }

        private void SetupBobot_Load(object sender, EventArgs e)
        {
            string connStr = @"Data Source=.\SQLEXPRESS;Integrated Security=SSPI;Initial
            Catalog=Db_Skripsi";
            conn = new SqlConnection(connStr);
            modeLoad();
        }

        private double GetBobot(int kriteria)
        {
            double bobotx = 0;
            string strSql = @"select bobot from kriteria where kd_kriteria=@kd_kriteria";
            cmd = new SqlCommand(strSql, conn);
            cmd.Parameters.AddWithValue("@kd_kriteria", kriteria);
            conn.Close();
            conn.Open();
            dr = cmd.ExecuteReader();
            dr.Read();
            bobotx = Convert.ToDouble(dr["bobot"]);
            return bobotx;
        }

        private void isiTextBox()
        {
            txt1.Text = GetBobot(1).ToString();
```

```
    txt2.Text = GetBobot(2).ToString();
    txt3.Text = GetBobot(3).ToString();
    txt4.Text = GetBobot(4).ToString();
    txt5.Text = GetBobot(5).ToString();
    txt6.Text = (GetBobot(1) + GetBobot(2) + GetBobot(3) + GetBobot(4) +
   GetBobot(5)).ToString();
}

private void isiJumlah()
{
    if (txt1.Text != "" && txt2.Text != "" && txt3.Text != "" && txt4.Text != "" &&
   txt5.Text != "")
    {
        txt6.Text = (Convert.ToInt32(txt1.Text) + Convert.ToInt32(txt2.Text) +
        Convert.ToInt32(txt3.Text) + Convert.ToInt32(txt4.Text) +
        Convert.ToInt32(txt5.Text)).ToString();
    }
}
private void modeLoad()
{
    txt1.Enabled = false;
    txt2.Enabled = false;
    txt3.Enabled = false;
    txt4.Enabled = false;
    txt5.Enabled = false;
    txt6.Enabled = false;
    btnSimpan.Visible = false;
    btnBatal.Visible = false;
    btnUbah.Visible = true;
    btnKeluar.Visible = true;
    isiTextBox();
}

private void modeUbah()
{
    txt1.Enabled = true;
    txt2.Enabled = true;
    txt3.Enabled = true;
    txt4.Enabled = true;
    txt5.Enabled = true;
    txt6.Enabled = false;
    btnSimpan.Visible = true;
    btnBatal.Visible = true;
    btnUbah.Visible = false;
    btnKeluar.Visible = false;
}

private void btnUbah_Click(object sender, EventArgs e)
{
    modeUbah();
}

private void btnKeluar_Click(object sender, EventArgs e)
{
    this.Close();
}

private void btnSimpan_Click(object sender, EventArgs e)
{
    int number2;
    if (txt1.Text == "" || txt2.Text == "" || txt3.Text == "" || txt4.Text == "" ||
   txt5.Text == "")
    {
        MessageBox.Show("Lengkapi semua data", "Perhatian", MessageBoxButtons.OK,
        MessageBoxIcon.Stop, MessageBoxDefaultButton.Button1);
    }
```

```
    else if (int.TryParse(txt1.Text, out number2) == false ||
    int.TryParse(txt2.Text, out number2) == false || int.TryParse(txt3.Text, out
    number2) == false || int.TryParse(txt4.Text, out number2) == false ||
    int.TryParse(txt5.Text, out number2) == false)
    {
        MessageBox.Show("Masukkan bobot harus berupa angka", "Perhatian",
        MessageBoxButtons.OK, MessageBoxIcon.Stop,
        MessageBoxDefaultButton.Button1);
    }
    else if (Convert.ToInt32(txt6.Text) > 100)
    {
        MessageBox.Show("Jumlah bobot tidak boleh > 100", "Perhatian",
        MessageBoxButtons.OK, MessageBoxIcon.Stop,
        MessageBoxDefaultButton.Button1);
    }
    else
    {
        string strSql = @"update kriteria set Bobot =
                        case when(Kd_Kriteria = 1) then @bobot1
                            when(Kd_Kriteria = 2) then @bobot2
                            when(Kd_Kriteria = 3) then @bobot3
                            when(Kd_Kriteria = 4) then @bobot4
                        when(Kd_Kriteria = 5) then @bobot5
                            else 0 end";
        cmd = new SqlCommand(strSql, conn);
        cmd.Parameters.AddWithValue("@bobot1", Convert.ToInt32(txt1.Text));
        cmd.Parameters.AddWithValue("@bobot2", Convert.ToInt32(txt2.Text));
        cmd.Parameters.AddWithValue("@bobot3", Convert.ToInt32(txt3.Text));
        cmd.Parameters.AddWithValue("@bobot4", Convert.ToInt32(txt4.Text));
        cmd.Parameters.AddWithValue("@bobot5", Convert.ToInt32(txt5.Text));
        conn.Close();
        conn.Open();
        cmd.ExecuteNonQuery();
        MessageBox.Show("Bobot berhasil diubah", "Konfirmasi",
        MessageBoxButtons.OK, MessageBoxIcon.Information,
        MessageBoxDefaultButton.Button1);
        this.Close();
    }
}

private void btnBatal_Click(object sender, EventArgs e)
{
    modeLoad();
}

private void txt1_Leave(object sender, EventArgs e)
{
     isiJumlah();
}

private void txt2_Leave(object sender, EventArgs e)
{
    isiJumlah();
}

private void txt3_Leave(object sender, EventArgs e)
{
    isiJumlah();
}

private void txt4_Leave(object sender, EventArgs e)
{
    isiJumlah();
}

private void txt5_Leave(object sender, EventArgs e)
```

```
        {
            isiJumlah();
        }

        private void txt1_KeyPress(object sender, KeyPressEventArgs e)
        {
            if (!char.IsControl(e.KeyChar) && !char.IsDigit(e.KeyChar))
            {
                e.Handled = true;
            }
        }

        private void txt2_KeyPress(object sender, KeyPressEventArgs e)
        {
            if (!char.IsControl(e.KeyChar) && !char.IsDigit(e.KeyChar))
            {
                e.Handled = true;
            }
        }

        private void txt3_KeyPress(object sender, KeyPressEventArgs e)
        {
            if (!char.IsControl(e.KeyChar) && !char.IsDigit(e.KeyChar))
            {
                e.Handled = true;
            }
        }

        private void txt4_KeyPress(object sender, KeyPressEventArgs e)
        {
            if (!char.IsControl(e.KeyChar) && !char.IsDigit(e.KeyChar))
            {
                e.Handled = true;
            }
        }

        private void txt5_KeyPress(object sender, KeyPressEventArgs e)
        {
            if (!char.IsControl(e.KeyChar) && !char.IsDigit(e.KeyChar))
            {
                e.Handled = true;
            }
        }

    }
}
```


Form Pengaturan Pengguna

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Data.SqlClient;
using System.Configuration;

namespace Skripsi0501
{
    public partial class SetupUser : Form
    {
```

```
private SqlDataAdapter da;
private SqlCommandBuilder cb; //men-generate perintah insert update delete
private DataSet ds;
private BindingSource bs;
private SqlConnection conn;
private SqlCommand cmd;
private SqlDataReader sdr;
Boolean baru;
string kodelama;

public SetupUser()
{
    InitializeComponent();
}

private void SetupUser_Load(object sender, EventArgs e)
{
    string connStr = @"Data Source=.\SQLEXPRESS;Integrated Security=SSPI;Initial
   Catalog=Db_Skripsi";
    conn = new SqlConnection(connStr);

    modeSave();
    IsiDataSet();
    TambahBinding();
}

#region mode
private void modeSave()
{
    txtUsername.Enabled = false;
    txtPass.Enabled = false;
    cmbHak.Enabled = false;

    btnTop.Enabled = true;
    btnPrev.Enabled = true;
    btnNext.Enabled = true;
    btnEnd.Enabled = true;
    btnNew.Enabled = true;
    btnEdit.Enabled = true;
    btnSave.Enabled = false;
    btnUndo.Enabled = false;
    btnDel.Enabled = true;
    btnClose.Enabled = true;
    dgvUser.Enabled = true;
}

private void modeNew()
{
    txtUsername.Enabled = true;
    txtPass.Enabled = true;
    cmbHak.Enabled = true;

    btnTop.Enabled = false;
    btnPrev.Enabled = false;
    btnNext.Enabled = false;
    btnEnd.Enabled = false;
    btnNew.Enabled = false;
    btnEdit.Enabled = false;
    btnSave.Enabled = true;
    btnUndo.Enabled = true;
    btnDel.Enabled = false;
    btnClose.Enabled = true;
    dgvUser.Enabled = false;
}
#endregion
```

```
private void IsiDataSet()
{
    string strSql = @"select * from Users";
    cmd = new SqlCommand(strSql, conn);
    conn.Close();
    conn.Open();

    da = new SqlDataAdapter();
    cb = new SqlCommandBuilder(da);
    da.SelectCommand = cmd;
    ds = new DataSet();
    da.Fill(ds, "Users");
    bs = new BindingSource();
    DataTable dt = ds.Tables["users"];
    var qUserEnum = dt.AsEnumerable();
    bs.DataSource = from m in qUserEnum
                    select new
                    {
                        username = m["username"].ToString(),
                        sandi = m["sandi"].ToString(),
                        hak = m["hak"].ToString()
                    };
    dgvUser.DataSource = bs;

    dgvUser.Columns[0].HeaderCell.Value = "Nama Pengguna";
    dgvUser.Columns[1].HeaderCell.Value = "Kata Sandi";
    dgvUser.Columns[2].HeaderCell.Value = "Hak Akses";
    dgvUser.Columns[0].Width = 139;
    dgvUser.Columns[1].Width = 128;
    dgvUser.Columns[2].Width = 65;
}

private void TambahBinding()
{
    HapusBinding();
    txtUsername.DataBindings.Add("Text", bs, "username");
    txtPass.DataBindings.Add("Text", bs, "sandi");
    cmbHak.DataSource = bs;
    cmbHak.DisplayMember = "hak";
}

private void HapusBinding()
{
    txtUsername.DataBindings.Clear();
    txtPass.DataBindings.Clear();
    cmbHak.DataBindings.Clear();
}

private void SelectHak()
{
    cmbHak.DataSource = null;
    cmbHak.Items.Add("Admin");
    cmbHak.Items.Add("Pengguna");
}

private string KodeUser()
{
    string kd_user = "";
    string strSql = @"select top 1 Kd_User from users order by
    substring(Kd_User,2,1) desc";
    cmd = new SqlCommand(strSql, conn);
    conn.Close();
    conn.Open();

    sdr = cmd.ExecuteReader();
    if (sdr.HasRows)
```

```
    {
        while (sdr.Read())
        {
            kd_user = sdr["kd_user"].ToString();
        }
    }
    sdr.Close();
    kd_user = kd_user.Substring(1, 1); //ambil kode user 1 digit terakhir
    int kd_user_int = Convert.ToInt32(kd_user) + 1; // dijadikan integer trus
    tambahin 1
    kd_user = "00" + kd_user_int.ToString(); // ditambah dengan string 0000
    return kd_user = "U" + kd_user.Substring(kd_user.Length - 1, 1); // dipotong
    lagi ambil 1 digit paling belakang
}

private void UpdateUser()
{
    string strSql = @"update users set username=@username, sandi=@sandi, hak=@hak
    where kd_user=@kd_user";
    cmd = new SqlCommand(strSql, conn);
    cmd.Parameters.AddWithValue("@kd_user", kodelama);
    cmd.Parameters.AddWithValue("@username", txtUsername.Text);
    cmd.Parameters.AddWithValue("@sandi", txtPass.Text);
    cmd.Parameters.AddWithValue("@hak", cmbHak.SelectedItem.ToString());
    conn.Close();

    try
    {
        conn.Open();
        int status = cmd.ExecuteNonQuery();
        if (status == 1)
        {
            MessageBox.Show("Data berhasil diubah");
        }
    }
    catch (SqlException sqlEx)
    {
        MessageBox.Show("Error : " + sqlEx.Number.ToString() + " " +
        sqlEx.Message);
    }
    finally
    {
        cmd.Dispose();
        conn.Close();
    }
}

private void btnTop_Click(object sender, EventArgs e)
{
    bs.MoveFirst();
}

private void btnPrev_Click(object sender, EventArgs e)
{
    bs.MovePrevious();
}

private void btnNext_Click(object sender, EventArgs e)
{
    bs.MoveNext();
}

private void btnEnd_Click(object sender, EventArgs e)
{
    bs.MoveLast();
}
```

```
private void btnNew_Click(object sender, EventArgs e)
{
    modeNew();
    HapusBinding();
    txtUsername.Text = string.Empty;
    txtPass.Text = string.Empty;
    SelectHak();
    baru = true;
}

private void btnEdit_Click(object sender, EventArgs e)
{
    modeNew();
    baru = false;
    DataRow dro = ds.Tables["users"].Rows[bs.Position];
    kodelama = dro["kd_user"].ToString();
    string haklama = dro["hak"].ToString();

    SelectHak();
    cmbHak.SelectedItem = haklama;
}

private void btnSave_Click(object sender, EventArgs e)
{
    string kodeuser = KodeUser();
    if (txtUsername.Text == "" || txtPass.Text == "" ||
    cmbHak.SelectedItem.ToString() == "")
    {
        MessageBox.Show("Lengkapi semua data", "Perhatian", MessageBoxButtons.OK,
        MessageBoxIcon.Stop, MessageBoxDefaultButton.Button1);
    }
    else
    {
        if (baru)
        {
            DataRow dro = ds.Tables["users"].NewRow();
            dro["kd_user"] = kodeuser;
            dro["username"] = txtUsername.Text;
            dro["sandi"] = txtPass.Text;
            dro["hak"] = cmbHak.SelectedItem;
            ds.Tables["users"].Rows.Add(dro);
            da.Update(ds.Tables["users"]);

            MessageBox.Show("Data berhasil masuk");
        }
        else
        {
            UpdateUser();
        }
        modeSave();
        IsiDataSet();
        TambahBinding();
    }
}

private void btnUndo_Click(object sender, EventArgs e)
{
    modeSave();
    TambahBinding();
}

private void btnDel_Click(object sender, EventArgs e)
{
```

```
            if (MessageBox.Show("Yakin menghapus?", "Konfirmasi", MessageBoxButtons.YesNo,
            MessageBoxIcon.Question, MessageBoxDefaultButton.Button2) ==
            System.Windows.Forms.DialogResult.Yes)
            {
                DataTable dt = ds.Tables["Users"];
                DataRow dro = dt.Rows[bs.Position];
                dro.Delete();
                da.Update(dt);

                MessageBox.Show("Data berhasil dihapus");
                modeSave();
                IsiDataSet();
                TambahBinding();
            }
        }

        private void btnClose_Click(object sender, EventArgs e)
        {
            this.Close();
        }
    }
}
```

Form Proses Tabel Resume Kota

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Data.SqlClient;
using System.Configuration;
using System.Diagnostics;
using System.IO;

namespace Skripsi0501
{
    public partial class ProsesResume : Form
    {
        private SqlDataAdapter da;
        private SqlCommandBuilder cb; //men-generate perintah insert update delete
        private DataSet ds;
        private BindingSource bs;
        private SqlConnection conn;
        private SqlCommand cmd;
        private SqlDataReader dr;
        string nmx = "";
        string je = "";
        string kd_kota = "";

        List<Kota> lstKota = new List<Kota>();
        List<Peningkatan> lstPen = new List<Peningkatan>();
        List<Resume> lstResume = new List<Resume>();

        public ProsesResume()
        {
            InitializeComponent();
        }

        private void ProsesPeningkatan_Load(object sender, EventArgs e)
        {
```

```
    string connStr = @"Data Source=.\SQLEXPRESS;Integrated Security=SSPI;Initial
    Catalog=Db_Skripsi";
    conn = new SqlConnection(connStr);
    conn.Open();
    cmbCari.SelectedItem = "Kota";
    chart1.Titles.Add("Jumlah Mahasiswa Pertahun");
    IsiDgvKota();
}

private void IsiDgvKota()
{
    string strSql = @"select nm_prop,jenis,kota.nm_kota as nmKota,
    round(resume.peningkatan,2) as Pen,rr_meningkat, rr_menurun,resume.jum_mhs as
    jumMhs,resume.jarak,resume.jum_sma as jumSma,round(resume.rr_ipk,2) as rrIpk
    from resume, kota, propinsi, peningkatan where resume.kd_kota = kota.kd_kota
    and kota.kd_prop=propinsi.kd_prop and resume.kd_kota = peningkatan.kd_kota
    order by propinsi.nm_prop";
    cmd = new SqlCommand(strSql, conn);
    conn.Close();
    conn.Open();
    da = new SqlDataAdapter();
    cb = new SqlCommandBuilder(da);
    da.SelectCommand = cmd;
    ds = new DataSet();
    da.Fill(ds, "hasilakhir");
    bs = new BindingSource();
    bs.DataSource = ds.Tables["hasilakhir"];
    dgvKota.DataSource = bs;
    StyleGrid();

}

private void btnCari_Click(object sender, EventArgs e)
{
    if (cmbCari.SelectedItem.ToString() == "Kota")
    {

        string strSql = @"select nm_prop, jenis, kota.nm_kota, resume.peningkatan,
        rr_meningkat, rr_menurun, resume.jum_mhs, resume.jarak, resume.jum_sma,
        round(resume.rr_ipk,2) from resume, kota, propinsi, peningkatan where
        resume.kd_kota = kota.kd_kota and kota.kd_prop=propinsi.kd_prop and
        kota.nm_kota like '%' + @nm_kota + '%' and
        resume.kd_kota=peningkatan.kd_kota order by propinsi.nm_prop";
        cmd = new SqlCommand(strSql, conn);
        cmd.Parameters.AddWithValue("@nm_kota", txtCari.Text.Trim());
    }
    else if (cmbCari.SelectedItem.ToString() == "Propinsi")
    {
        string strSql = @"select nm_prop, jenis, kota.nm_kota, resume.peningkatan,
        rr_meningkat, rr_menurun, resume.jum_mhs, resume.jarak, resume.jum_sma,
        round(resume.rr_ipk,2) from resume, kota, propinsi, peningkatan where
        resume.kd_kota = kota.kd_kota and kota.kd_prop=propinsi.kd_prop and nm_prop
        like '%' + @nm_prop + '%' and resume.kd_kota=peningkatan.kd_kota order by
        propinsi.nm_prop";
        cmd = new SqlCommand(strSql, conn);
        cmd.Parameters.AddWithValue("@nm_prop", txtCari.Text.Trim());
    }
    else if (cmbCari.SelectedItem.ToString() == "Rata-rata Naik > Rata-rata Turun")
    {
        string strSql = @"select nm_prop, jenis, kota.nm_kota,
        round(resume.peningkatan,2), rr_meningkat, rr_menurun, resume.jum_mhs,
        resume.jarak, resume.jum_sma,round(resume.rr_ipk,2) from resume, kota,
        propinsi, peningkatan where resume.kd_kota = kota.kd_kota and
        kota.kd_prop=propinsi.kd_prop and resume.kd_kota=peningkatan.kd_kota and
```

```
            peningkatan.rr_meningkat>(peningkatan.rr_menurun*-1) order by
            propinsi.nm_prop";
            cmd = new SqlCommand(strSql, conn);
        }
        else if (cmbCari.SelectedItem.ToString() == "Rata-rata Naik < Rata-rata Turun")
        {
            string strSql = @"select nm_prop, jenis, kota.nm_kota,
            round(resume.peningkatan,2), rr_meningkat, rr_menurun, resume.jum_mhs,
            resume.jarak,resume.jum_sma,round(resume.rr_ipk,2) from resume, kota,
            propinsi, peningkatan where resume.kd_kota = kota.kd_kota and
            kota.kd_prop=propinsi.kd_prop and resume.kd_kota=peningkatan.kd_kota and
            peningkatan.rr_meningkat<(peningkatan.rr_menurun*-1) order by
            propinsi.nm_prop";
            cmd = new SqlCommand(strSql, conn);
        }
        else if (cmbCari.SelectedItem.ToString() == "Rata-rata Naik = Rata-rata Turun")
        {
            string strSql = @"select nm_prop, jenis, kota.nm_kota,
            round(resume.peningkatan,2),rr_meningkat,rr_menurun,resume.jum_mhs,
            resume.jarak, resume.jum_sma,round(resume.rr_ipk,2) from resume, kota,
            propinsi, peningkatan where resume.kd_kota = kota.kd_kota and
            kota.kd_prop=propinsi.kd_prop and resume.kd_kota=peningkatan.kd_kota and
            peningkatan.rr_meningkat=(peningkatan.rr_menurun*-1) order by
            propinsi.nm_prop";
            cmd = new SqlCommand(strSql, conn);
        }
        conn.Close();
        conn.Open();
        da = new SqlDataAdapter();
        cb = new SqlCommandBuilder(da);
        da.SelectCommand = cmd;
        ds = new DataSet();
        da.Fill(ds, "hasilakhir");
        bs = new BindingSource();
        bs.DataSource = ds.Tables["hasilakhir"];
        dgvKota.DataSource = bs;
        StyleGrid();
        dgvKota.Focus();
        Chart();
        DaftarSMA();
    }

    private void dgvKota_CellMouseClick(object sender, DataGridViewCellMouseEventArgs
     e)
    {
        if (e.RowIndex == 0 || e.RowIndex > 0)
        {
            nmx = dgvKota.Rows[e.RowIndex].Cells[2].FormattedValue.ToString();
            je = dgvKota.Rows[e.RowIndex].Cells[1].FormattedValue.ToString();
            string strSql = @"select kd_kota from kota where nm_kota=@nm_kota and
            jenis=@jenis";
            cmd = new SqlCommand(strSql, conn);
            cmd.Parameters.AddWithValue("@nm_kota", nmx);
            cmd.Parameters.AddWithValue("@jenis", je);
            conn.Close();
            conn.Open();
            dr = cmd.ExecuteReader();
            if (dr.HasRows)
            {
                while (dr.Read())
                {
                    kd_kota = dr["kd_kota"].ToString();
                }
            }
        }
        Chart();
```

```
    DaftarSMA();
}

private void dgvKota_RowEnter(object sender, DataGridViewCellEventArgs e)
{
    if (e.RowIndex == 0 || e.RowIndex > 0)
    {
        nmx = dgvKota.Rows[e.RowIndex].Cells[2].FormattedValue.ToString();
        je = dgvKota.Rows[e.RowIndex].Cells[1].FormattedValue.ToString();

        string strSql = @"select kd_kota from kota where nm_kota=@nm_kota and
        jenis=@jenis";
        cmd = new SqlCommand(strSql, conn);
        cmd.Parameters.AddWithValue("@nm_kota", nmx);
        cmd.Parameters.AddWithValue("@jenis", je);
        conn.Close();
        conn.Open();
        dr = cmd.ExecuteReader();
        if (dr.HasRows)
        {
            while (dr.Read())
            {
                kd_kota = dr["kd_kota"].ToString();
            }
        }
    }
    Chart();
    DaftarSMA();
}

private void Chart()
{
    chart1.Titles.RemoveAt(0);
    chart1.Series[("Tahun")].Points.Clear();
    chart1.Titles.Add("Jumlah Mahasiswa Pertahun");
    string strSql = @"select tahun1, tahun2, tahun3, tahun4, tahun5 from
    peningkatan where kd_kota=@kd_kota";
    cmd = new SqlCommand(strSql, conn);
    cmd.Parameters.AddWithValue("@kd_kota", kd_kota);
    conn.Close();
    conn.Open();

    dr = cmd.ExecuteReader();
    if (dr.HasRows)
    {
        while (dr.Read())
        {
            chart1.Series[("Tahun")].Points.AddY(Convert.ToInt32(dr["tahun1"]));
            chart1.Series[("Tahun")].Points.AddY(Convert.ToInt32(dr["tahun2"]));
            chart1.Series[("Tahun")].Points.AddY(Convert.ToInt32(dr["tahun3"]));
            chart1.Series[("Tahun")].Points.AddY(Convert.ToInt32(dr["tahun4"]));
            chart1.Series[("Tahun")].Points.AddY(Convert.ToInt32(dr["tahun5"]));
        }
    }
}

private void DaftarSMA()
{
    string strSql = @"select distinct(sma) from mahasiswa where kd_kota=@kd_kota";
    cmd = new SqlCommand(strSql, conn);
    cmd.Parameters.AddWithValue("@kd_kota", kd_kota);
    conn.Close();
    conn.Open();

    da = new SqlDataAdapter();
    cb = new SqlCommandBuilder(da);
```

```
    da.SelectCommand = cmd;
    ds = new DataSet();
    da.Fill(ds, "sma");
    bs = new BindingSource();
    bs.DataSource = ds.Tables["sma"];
    dgvSma.DataSource = bs;

    dgvSma.Columns[0].HeaderCell.Value = "Daftar SMA";
    dgvSma.Columns[0].Width = 260;
}

private void StyleGrid()
{
    dgvKota.Columns[0].HeaderCell.Value = "Propinsi";
    dgvKota.Columns[1].HeaderCell.Value = "Jenis";
    dgvKota.Columns[2].HeaderCell.Value = "Kota";
    dgvKota.Columns[3].HeaderCell.Value = "Peningkatan";
    dgvKota.Columns[4].HeaderCell.Value = "Rata2 Naik";
    dgvKota.Columns[5].HeaderCell.Value = "Rata2 Turun";
    dgvKota.Columns[6].HeaderCell.Value = "Jum Mhs";
    dgvKota.Columns[7].HeaderCell.Value = "Jarak";
    dgvKota.Columns[8].HeaderCell.Value = "Jum SMA";
    dgvKota.Columns[9].HeaderCell.Value = "Rata2 IPK";
    dgvKota.Columns[0].Width = 140;
    dgvKota.Columns[1].Width = 53;
    dgvKota.Columns[2].Width = 140;
    dgvKota.Columns[3].Width = 77;
    dgvKota.Columns[4].Width = 43;
    dgvKota.Columns[5].Width = 43;
    dgvKota.Columns[6].Width = 40;
    dgvKota.Columns[7].Width = 40;
    dgvKota.Columns[8].Width = 40;
    dgvKota.Columns[9].Width = 43;
    textBox1.Text = dgvKota.BindingContext[dgvKota.DataSource].Count.ToString();
}

private void btnKeluar_Click(object sender, EventArgs e)
{
    this.Close();
}

#region analisis
//Tahun awal > tahun akhir (abaikan 0)
private void AnalisisSatu()
{
    int itung = 0;
    for (int x = 0; x < lstPen.Count; x++)
    {
        if (lstPen[x].tahun1 != 0)
        {
            if (lstPen[x].tahun5 != 0)
            {
                if (lstPen[x].tahun1 > lstPen[x].tahun5)
                    itung++;
            }
            else if (lstPen[x].tahun4 != 0)
            {
                if (lstPen[x].tahun1 > lstPen[x].tahun4)
                    itung++;
            }
            else if (lstPen[x].tahun3 != 0)
            {
                if (lstPen[x].tahun1 > lstPen[x].tahun3)
                    itung++;
            }
            else if (lstPen[x].tahun2 != 0)
```

```
            {
                if (lstPen[x].tahun1 > lstPen[x].tahun2)
                    itung++;
            }
        }
        else if (lstPen[x].tahun2 != 0)
        {
            if (lstPen[x].tahun5 != 0)
            {
                if (lstPen[x].tahun2 > lstPen[x].tahun5)
                    itung++;
            }
            else if (lstPen[x].tahun4 != 0)
            {
                if (lstPen[x].tahun2 > lstPen[x].tahun4)
                    itung++;
            }
            else if (lstPen[x].tahun3 != 0)
            {
                if (lstPen[x].tahun2 > lstPen[x].tahun3)
                    itung++;
            }
        }
        else if (lstPen[x].tahun3 != 0)
        {
            if (lstPen[x].tahun5 != 0)
            {
                if (lstPen[x].tahun3 > lstPen[x].tahun5)
                    itung++;
            }
            else if (lstPen[x].tahun4 != 0)
            {
                if (lstPen[x].tahun3 > lstPen[x].tahun4)
                    itung++;
            }
        }
        else if (lstPen[x].tahun4 != 0)
        {
            if (lstPen[x].tahun5 != 0)
            {
                if (lstPen[x].tahun4 > lstPen[x].tahun5)
                    itung++;
            }
        }
    }
    MessageBox.Show("Tahun awal > Tahun Akhir : " + itung.ToString());
}

//Tahun awal < tahun akhir (abaikan 0)
private void AnalisisDua()
{
    int itung = 0;
    for (int x = 0; x < lstPen.Count; x++)
    {
        if (lstPen[x].tahun1 != 0)
        {
            if (lstPen[x].tahun5 != 0)
            {
                if (lstPen[x].tahun1 < lstPen[x].tahun5)
                    itung++;
            }
            else if (lstPen[x].tahun4 != 0)
            {
                if (lstPen[x].tahun1 < lstPen[x].tahun4)
                    itung++;
            }
```

```
            else if (lstPen[x].tahun3 != 0)
            {
                if (lstPen[x].tahun1 < lstPen[x].tahun3)
                    itung++;
            }
            else if (lstPen[x].tahun2 != 0)
            {
                if (lstPen[x].tahun1 < lstPen[x].tahun2)
                    itung++;
            }
        }
        else if (lstPen[x].tahun2 != 0)
        {
            if (lstPen[x].tahun5 != 0)
            {
                if (lstPen[x].tahun2 < lstPen[x].tahun5)
                    itung++;
            }
            else if (lstPen[x].tahun4 != 0)
            {
                if (lstPen[x].tahun2 < lstPen[x].tahun4)
                    itung++;
            }
            else if (lstPen[x].tahun3 != 0)
            {
                if (lstPen[x].tahun2 < lstPen[x].tahun3)
                    itung++;
            }
        }
        else if (lstPen[x].tahun3 != 0)
        {
            if (lstPen[x].tahun5 != 0)
            {
                if (lstPen[x].tahun3 < lstPen[x].tahun5)
                    itung++;
            }
            else if (lstPen[x].tahun4 != 0)
            {
                if (lstPen[x].tahun3 < lstPen[x].tahun4)
                    itung++;
            }
        }
        else if (lstPen[x].tahun4 != 0)
        {
            if (lstPen[x].tahun5 != 0)
            {
                if (lstPen[x].tahun4 < lstPen[x].tahun5)
                    itung++;
            }
        }
    }
    MessageBox.Show("Tahun Awal < Tahun Akhir : " + itung.ToString());

}

//Tahun awal = tahun akhir (abaikan 0)
private void AnalisisTiga()
{
    int itung = 0;
    for (int x = 0; x < lstPen.Count; x++)
    {
        if (lstPen[x].tahun1 != 0)
        {
            if (lstPen[x].tahun5 != 0)
            {
                if (lstPen[x].tahun1 == lstPen[x].tahun5)
```

```
                    itung++;
                }
                else if (lstPen[x].tahun4 != 0)
                {
                    if (lstPen[x].tahun1 == lstPen[x].tahun4)
                        itung++;
                }
                else if (lstPen[x].tahun3 != 0)
                {
                    if (lstPen[x].tahun1 == lstPen[x].tahun3)
                        itung++;
                }
                else if (lstPen[x].tahun2 != 0)
                {
                    if (lstPen[x].tahun1 == lstPen[x].tahun2)
                        itung++;
                }
            }
            else if (lstPen[x].tahun2 != 0)
            {
                if (lstPen[x].tahun5 != 0)
                {
                    if (lstPen[x].tahun2 == lstPen[x].tahun5)
                        itung++;
                }
                else if (lstPen[x].tahun4 != 0)
                {
                    if (lstPen[x].tahun2 == lstPen[x].tahun4)
                        itung++;
                }
                else if (lstPen[x].tahun3 != 0)
                {
                    if (lstPen[x].tahun2 == lstPen[x].tahun3)
                        itung++;
                }
            }
            else if (lstPen[x].tahun3 != 0)
            {
                if (lstPen[x].tahun5 != 0)
                {
                    if (lstPen[x].tahun3 == lstPen[x].tahun5)
                        itung++;
                }
                else if (lstPen[x].tahun4 != 0)
                {
                    if (lstPen[x].tahun3 == lstPen[x].tahun4)
                        itung++;
                }
            }
            else if (lstPen[x].tahun4 != 0)
            {
                if (lstPen[x].tahun5 != 0)
                {
                    if (lstPen[x].tahun4 == lstPen[x].tahun5)
                        itung++;
                }
            }
        }
        MessageBox.Show("Tahun Awal = Tahun Akhir : " + itung.ToString());

    }

    //Tiap tahun menurun
    private void AnalisisEmpat()
    {
        int itung = 0;
```

```
    for (int x = 0; x < lstPen.Count; x++)
    {
        if (lstPen[x].tahun1 != 0 && lstPen[x].tahun2 != 0 && lstPen[x].tahun3 != 0
        && lstPen[x].tahun4 != 0 && lstPen[x].tahun5 != 0)
        {
            if (lstPen[x].tahun1 > lstPen[x].tahun2 && lstPen[x].tahun2 >
            lstPen[x].tahun3 && lstPen[x].tahun3 > lstPen[x].tahun4 &&
            lstPen[x].tahun4 > lstPen[x].tahun5)
                itung++;
        }
    }
    MessageBox.Show("Tiap tahun menurun: " + itung.ToString());
}

//Tiap tahun meningkat
private void AnalisisLima()
{
    int itung = 0;
    for (int x = 0; x < lstPen.Count; x++)
    {
        if (lstPen[x].tahun1 != 0 && lstPen[x].tahun2 != 0 && lstPen[x].tahun3 != 0
        && lstPen[x].tahun4 != 0 && lstPen[x].tahun5 != 0)
        {
            if (lstPen[x].tahun1 < lstPen[x].tahun2 && lstPen[x].tahun2 <
            lstPen[x].tahun3 && lstPen[x].tahun3 < lstPen[x].tahun4 &&
            lstPen[x].tahun4 < lstPen[x].tahun5)
                itung++;
        }
    }
    MessageBox.Show("Tiap tahun meningkat: " + itung.ToString());
}

//Tiap tahun sama
private void AnalisisEnam()
{
    int itung = 0;
    for (int x = 0; x < lstPen.Count; x++)
    {
        if (lstPen[x].tahun1 != 0 && lstPen[x].tahun2 != 0 && lstPen[x].tahun3 != 0
        && lstPen[x].tahun4 != 0 && lstPen[x].tahun5 != 0)
        {
            if (lstPen[x].tahun1 == lstPen[x].tahun2 && lstPen[x].tahun2 ==
            lstPen[x].tahun3 && lstPen[x].tahun3 == lstPen[x].tahun4 &&
            lstPen[x].tahun4 == lstPen[x].tahun5)
                itung++;
        }
    }
    MessageBox.Show("Tiap tahun sama: " + itung.ToString());
}
#endregion

private void cmbCari_SelectedIndexChanged(object sender, EventArgs e)
{
    if (cmbCari.SelectedItem.ToString() == "Kota" ||
    cmbCari.SelectedItem.ToString() == "Propinsi")
    {
        txtCari.Enabled = true;
    }
    else if (cmbCari.SelectedItem.ToString() == "Rata-rata Naik > Rata-rata Turun"
        || cmbCari.SelectedItem.ToString() == "Rata-rata Naik < Rata-rata Turun"
        || cmbCari.SelectedItem.ToString() == "Rata-rata Naik = Rata-rata Turun")
    {
        txtCari.Enabled = false;
    }

}
```

```
    }
}
```

Form Penyaringan Kota

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Data.SqlClient;
using System.Configuration;

namespace Skripsi0501
{
    public partial class PenyaringanKota : Form
    {
        private SqlDataAdapter da;
        private SqlCommandBuilder cb; //men-generate perintah insert update delete
        private DataSet ds;
        private BindingSource bs;
        private SqlConnection conn;
        private SqlCommand cmd;
        private SqlDataReader dr;

        //pake variabel globar biar bisa dibaca pas Alert Show
        string cmbJarakStr = "";
        string cmbRerataIpkStr = "";
        string cmbPerbandinganStr = "";
        string cmbTahunStr = "";

        AlertForm alert;

        string nmx = "";
        string je = "";

        string kd_kota = "";
        List<Hasil> lstHasil = new List<Hasil>();

        public PenyaringanKota()
        {
            InitializeComponent();
        }

        private void PencarianKota_Load(object sender, EventArgs e)
        {
            string connStr = @"Data Source=.\SQLEXPRESS;Integrated Security=SSPI;Initial
            Catalog=Db_Skripsi";
            conn = new SqlConnection(connStr);

            MaksMin();

            cmbPerbandingan.SelectedIndex = 0;
            cmbTahun.SelectedIndex = 0;

            chart1.Titles.Add("Jumlah Mahasiswa Pertahun");
            backgroundWorker1.WorkerReportsProgress = true;
            backgroundWorker1.WorkerSupportsCancellation = true;
            //SetupBobot setupBbt = new SetupBobot();
            //setupBbt.ShowDialog();
```

```
}

private void HapusHasil()
{
    string strSql = @"delete from hasil";
    cmd = new SqlCommand(strSql,conn);
    conn.Close();
    conn.Open();
    cmd.ExecuteNonQuery();
}

private void CopyKodeKota()
{
    string strSql = @"insert into hasil(kd_kota) select resume.kd_kota from
   resume";
    cmd = new SqlCommand(strSql,conn);
    conn.Close();
    conn.Open();
    cmd.ExecuteNonQuery();
}

private double GetBobot(int kriteria)
{
    double bobotx = 0;
    string strSql = @"select bobot from kriteria where kd_kriteria=@kd_kriteria";
    cmd = new SqlCommand(strSql, conn);
    cmd.Parameters.AddWithValue("@kd_kriteria", kriteria);
    conn.Close();
    conn.Open();
    dr = cmd.ExecuteReader();
    dr.Read();
    bobotx = Convert.ToDouble(dr["bobot"])/100;
    return bobotx;
}

private void Peningkatan()
{
    double bobotx = GetBobot(1);
    double rangeawal = 0; double rangeakhir = 0;
    int nilai1 = 30; int nilai2 = 100; int nilai3 = 60;
    rangeawal = Convert.ToDouble(txtAwalPen.Text.Trim());
    rangeakhir= Convert.ToDouble(txtAkhirPen.Text.Trim());
    string strSql = @"update hasil set peningkatan= case when(kd_kota in (select
                    kd_kota from resume where peningkatan < @range_awal)) then
                    @nilai1 * @bobot when(kd_kota in (select kd_kota from resume
                    where peningkatan >=@range_awal and peningkatan
                    <=@range_akhir)) then @nilai2 * @bobot else @nilai3 * @bobot
                    end";
    cmd = new SqlCommand(strSql, conn);
    cmd.Parameters.AddWithValue("@range_awal", rangeawal);
    cmd.Parameters.AddWithValue("@range_akhir", rangeakhir);
    cmd.Parameters.AddWithValue("@nilai1", nilai1);
    cmd.Parameters.AddWithValue("@nilai2", nilai2);
    cmd.Parameters.AddWithValue("@nilai3", nilai3);
    cmd.Parameters.AddWithValue("@bobot", bobotx);
    conn.Close();
    conn.Open();
    cmd.ExecuteNonQuery();
}

private void JumlahMhs()
{
    double bobotx = GetBobot(2);
    int rangeawal = Convert.ToInt32(txtJumMhsAwal.Text.Trim());
    int rangeakhir = Convert.ToInt32(txtJumMhsAkhir.Text.Trim());
    int nilai1 = 30; int nilai2 = 100; int nilai3 = 60;
```

```
    string strSql = @"update hasil set jum_mhs= case when(kd_kota in (select
                    kd_kota from resume where jum_mhs < @range_awal)) then @nilai1
                    * @bobot when(kd_kota in (select kd_kota from resume where
                    jum_mhs >=@range_awal and jum_mhs <=@range_akhir)) then
                    @nilai2 * @bobot else @nilai3 * @bobot end";
    cmd = new SqlCommand(strSql, conn);
    cmd.Parameters.AddWithValue("@range_awal", rangeawal);
    cmd.Parameters.AddWithValue("@range_akhir", rangeakhir);
    cmd.Parameters.AddWithValue("@nilai1", nilai1);
    cmd.Parameters.AddWithValue("@nilai2", nilai2);
    cmd.Parameters.AddWithValue("@nilai3", nilai3);
    cmd.Parameters.AddWithValue("@bobot", bobotx);
    conn.Close();
    conn.Open();
    cmd.ExecuteNonQuery();
}

private void JarakKota()
{
    double bobotx = GetBobot(3);
    int nilai1 = 0; // lokal
    int nilai2 = 0; // dekat
    int nilai3 = 0; // jauh
    if (cmbJarakStr == "Lokal")
    {
        nilai1 = 100; nilai2 = 60; nilai3 = 30;
    }
    if (cmbJarakStr == "Dekat")
    {
        nilai1 = 60; nilai2 = 100; nilai3 = 30;
    }
    if (cmbJarakStr == "Jauh")
    {
        nilai1 = 30; nilai2 = 60; nilai3 = 100;
    }
    string strSql = @"update hasil set jarak= case when(kd_kota in (select kd_kota
                    from resume where jarak = 'lokal')) then @nilai1 * @bobot
                    when(kd_kota in (select kd_kota from resume where jarak =
                    'dekat')) then @nilai2 * @bobot else @nilai3 * @bobot end";
    cmd = new SqlCommand(strSql, conn);
    cmd.Parameters.AddWithValue("@nilai1", nilai1);
    cmd.Parameters.AddWithValue("@nilai2", nilai2);
    cmd.Parameters.AddWithValue("@nilai3", nilai3);
    cmd.Parameters.AddWithValue("@bobot", bobotx);
    conn.Close();
    conn.Open();
    cmd.ExecuteNonQuery();
}

private void JumlahSMA()
{
    double bobotx = GetBobot(4);
    int rangeawal = Convert.ToInt32(txtJumSmaAwal.Text.Trim());
    int rangeakhir = Convert.ToInt32(txtJumSmaAkhir.Text.Trim());
    int nilai1 = 30; int nilai2 = 100; int nilai3 = 60;

    string strSql = @"update hasil set jum_sma= case when(kd_kota in (select
                    kd_kota from resume where jum_sma < @range_awal)) then @nilai1
                    * @bobot when(kd_kota in (select kd_kota from resume where
                    jum_sma >=@range_awal and jum_sma <=@range_akhir)) then
                    @nilai2 * @bobot else @nilai3 * @bobot end";
    cmd = new SqlCommand(strSql, conn);
    cmd.Parameters.AddWithValue("@range_awal", rangeawal);
    cmd.Parameters.AddWithValue("@range_akhir", rangeakhir);
    cmd.Parameters.AddWithValue("@nilai1", nilai1);
```

```
    cmd.Parameters.AddWithValue("@nilai2", nilai2);
    cmd.Parameters.AddWithValue("@nilai3", nilai3);
    cmd.Parameters.AddWithValue("@bobot", bobotx);
    conn.Close();
    conn.Open();
    cmd.ExecuteNonQuery();
}

private void RerataIPK()
{
    double bobotx = GetBobot(5);
    int nilai1 = 0; int nilai2 = 0; int nilai3 = 0;
    if (cmbRerataIpkStr == "≥ 0  sampai  < 2.05")
    {
        nilai1 = 100; nilai2 = 60; nilai3 = 30;
    }
    if (cmbRerataIpkStr == "≥ 2.05  sampai  < 3")
    {
          nilai1 = 60; nilai2 = 100; nilai3 = 30;
    }
    if (cmbRerataIpkStr == "≥ 3  sampai  ≤ 4")
    {
        nilai1 = 30; nilai2 = 60; nilai3 = 100;
    }
    string strSql = @"update hasil set rr_ipk= case when(kd_kota in (select kd_kota
                    from resume where rr_ipk >= 0 and rr_ipk < 2.05)) then @nilai1
                    * @bobot when(kd_kota in (select kd_kota from resume where
                    rr_ipk >= 2.05 and rr_ipk < 3)) then @nilai2 * @bobot else
                    @nilai3 * @bobot end";
    cmd = new SqlCommand(strSql, conn);
    cmd.Parameters.AddWithValue("@nilai1", nilai1);
    cmd.Parameters.AddWithValue("@nilai2", nilai2);
    cmd.Parameters.AddWithValue("@nilai3", nilai3);
    cmd.Parameters.AddWithValue("@bobot", bobotx);
    conn.Close();
    conn.Open();
    cmd.ExecuteNonQuery();
}

private void IsiLstHasil()
{
    string strSql = @"select kd_kota, peningkatan, jum_mhs, jarak, jum_sma, rr_ipk
    from hasil";
    cmd = new SqlCommand(strSql, conn);
    conn.Close();
    conn.Open();
    dr = cmd.ExecuteReader();
    if (dr.HasRows)
    {
        while (dr.Read())
        {
            lstHasil.Add(new Hasil()
            {
                kd_kota = dr["kd_kota"].ToString(),
                peningkatan = Convert.ToInt32(dr["peningkatan"]),
                jum_mhs = Convert.ToInt32(dr["jum_mhs"]),
                jarak = Convert.ToInt32(dr["jarak"]),
                jum_sma = Convert.ToInt32(dr["jum_sma"]),
                rr_ipk = Convert.ToInt32(dr["rr_ipk"])
            });
        }
    }
}

private void JumlahNilai()
{
```

```
    int jum; string kd_kota;
    for (int x = 0; x < lstHasil.Count; x++)
    {
        kd_kota = lstHasil[x].kd_kota;
        jum = lstHasil[x].peningkatan + lstHasil[x].jum_mhs + lstHasil[x].jarak
              + lstHasil[x].jum_sma + lstHasil[x].rr_ipk;

        string strSql = @"update hasil set jumlah=@jumlah where kd_kota=@kd_kota";
        cmd = new SqlCommand(strSql, conn);
        cmd.Parameters.AddWithValue("@kd_kota", kd_kota);
        cmd.Parameters.AddWithValue("@jumlah", jum);
        conn.Close();
        conn.Open();
        cmd.ExecuteNonQuery();
    }
}

private void IsiDgvKota()
{
    //int jumSaring = Convert.ToInt32(txtJumSaring.Text.Trim());
    string strSql;
    if (cmbPerbandinganStr == "Rata-rata Naik > Rata-rata Turun")
    {
        if (cmbTahunStr == "Tidak Termasuk")
        {
            strSql = @"select top 25 nm_prop, jenis, kota.nm_kota,
            round(resume.peningkatan,2), rr_meningkat, rr_menurun, resume.jum_mhs,
            resume.jarak, resume.jum_sma, round(resume.rr_ipk,2) from hasil, kota,
            propinsi, resume, peningkatan where hasil.kd_kota=kota.kd_kota and
            resume.kd_kota = hasil.kd_kota and kota.kd_prop=propinsi.kd_prop and
            rr_meningkat > (rr_menurun*-1) and peningkatan.tahun1 <> 0 and
            peningkatan.tahun2 <> 0 and peningkatan.tahun3 <> 0 and
            peningkatan.tahun4 <> 0 and peningkatan.tahun5 <> 0 and
            peningkatan.kd_kota = resume.kd_kota order by hasil.jumlah desc";
            cmd = new SqlCommand(strSql, conn);
        }
        else
        {
            strSql = @"select top 25 nm_prop, jenis, kota.nm_kota,
            round(resume.peningkatan,2), rr_meningkat, rr_menurun, resume.jum_mhs,
            resume.jarak, resume.jum_sma, round(resume.rr_ipk,2) from hasil, kota,
            propinsi, resume, peningkatan where hasil.kd_kota=kota.kd_kota and
            resume.kd_kota = hasil.kd_kota and kota.kd_prop=propinsi.kd_prop and
            peningkatan.kd_kota = resume.kd_kota and rr_meningkat > (rr_menurun*-1)
            order by hasil.jumlah desc";
            cmd = new SqlCommand(strSql, conn);
        }
    }
    else if (cmbPerbandinganStr == "Rata-rata Naik < Rata-rata Turun")
    {
        if (cmbTahunStr == "Tidak Termasuk")
        {
            strSql = @"select top 25 nm_prop, jenis, kota.nm_kota,
            round(resume.peningkatan,2), rr_meningkat, rr_menurun, resume.jum_mhs,
            resume.jarak, resume.jum_sma, round(resume.rr_ipk,2) from hasil, kota,
            propinsi, resume, peningkatanwhere hasil.kd_kota=kota.kd_kota and
            resume.kd_kota = hasil.kd_kota and kota.kd_prop=propinsi.kd_prop and
            rr_meningkat < (rr_menurun*-1) and peningkatan.tahun1 <> 0 and
            peningkatan.tahun2 <> 0 and peningkatan.tahun3 <> 0 and
            peningkatan.tahun4 <> 0 and peningkatan.tahun5 <> 0 and
            peningkatan.kd_kota = resume.kd_kota order by hasil.jumlah desc";
            cmd = new SqlCommand(strSql, conn);
        }
        else
        {
```

```
            strSql = @"select top 25 nm_prop, jenis, kota.nm_kota,
            round(resume.peningkatan,2), rr_meningkat, rr_menurun, resume.jum_mhs,
            resume.jarak, resume.jum_sma, round(resume.rr_ipk,2) from hasil, kota,
            propinsi, resume, peningkatan where hasil.kd_kota=kota.kd_kota and
            resume.kd_kota = hasil.kd_kota and kota.kd_prop=propinsi.kd_prop and
            peningkatan.kd_kota = resume.kd_kota and rr_meningkat < (rr_menurun*-1)
            order by hasil.jumlah desc";
            cmd = new SqlCommand(strSql, conn);
        }
    }
    else if (cmbPerbandinganStr == "Rata-rata Naik = Rata-rata Turun")
    {
        if (cmbTahunStr == "Tidak Termasuk")
        {
            strSql = @"select top 25 nm_prop, jenis, kota.nm_kota,
            round(resume.peningkatan,2), rr_meningkat, rr_menurun, resume.jum_mhs,
            resume.jarak, resume.jum_sma, round(resume.rr_ipk,2) from hasil, kota,
            propinsi, resume, peningkatan where hasil.kd_kota=kota.kd_kota and
            resume.kd_kota = hasil.kd_kota and kota.kd_prop=propinsi.kd_prop and
            rr_meningkat = (rr_menurun*-1) and peningkatan.tahun1 <> 0 and
            peningkatan.tahun2 <> 0 and peningkatan.tahun3 <> 0 and
            peningkatan.tahun4 <> 0 and peningkatan.tahun5 <> 0 and
            peningkatan.kd_kota = resume.kd_kota order by hasil.jumlah desc";
            cmd = new SqlCommand(strSql, conn);
        }
        else
        {
            strSql = @"select top 25 nm_prop, jenis, kota.nm_kota,
            round(resume.peningkatan,2), rr_meningkat, rr_menurun, resume.jum_mhs,
            resume.jarak, resume.jum_sma, round(resume.rr_ipk,2) from hasil, kota,
            propinsi, resume, peningkatan where hasil.kd_kota=kota.kd_kota and
            resume.kd_kota = hasil.kd_kota and kota.kd_prop=propinsi.kd_prop and
            peningkatan.kd_kota = resume.kd_kota and rr_meningkat = (rr_menurun*-1)
            order by hasil.jumlah desc";
            cmd = new SqlCommand(strSql, conn);
        }
    }
    else if (cmbPerbandinganStr == "Tanpa Perbandingan")
    {
        if (cmbTahunStr == "Tidak Termasuk")
        {
            strSql = @"select top 25 nm_prop, jenis, kota.nm_kota,
            round(resume.peningkatan,2), rr_meningkat, rr_menurun, resume.jum_mhs,
            resume.jarak, resume.jum_sma, round(resume.rr_ipk,2) from hasil, kota,
            propinsi, resume, peningkatan where hasil.kd_kota=kota.kd_kota and
            resume.kd_kota = hasil.kd_kota and kota.kd_prop=propinsi.kd_prop and
            peningkatan.tahun1 <> 0 and peningkatan.tahun2 <> 0 and
            peningkatan.tahun3 <> 0 and peningkatan.tahun4 <> 0 and
            peningkatan.tahun5 <> 0 and peningkatan.kd_kota = resume.kd_kota order
            by hasil.jumlah desc";
            cmd = new SqlCommand(strSql, conn);
        }
        else
        {
            strSql = @"select top 25 nm_prop, jenis, kota.nm_kota,
            round(resume.peningkatan,2), rr_meningkat, rr_menurun, resume.jum_mhs,
            resume.jarak, resume.jum_sma, round(resume.rr_ipk,2) from hasil, kota,
            propinsi, resume, peningkatan where hasil.kd_kota=kota.kd_kota and
            resume.kd_kota = hasil.kd_kota and kota.kd_prop=propinsi.kd_prop and
            peningkatan.kd_kota = resume.kd_kota order by hasil.jumlah desc";
            cmd = new SqlCommand(strSql, conn);
        }
    }
    conn.Close();
    conn.Open();
```

```
        da = new SqlDataAdapter();
        cb = new SqlCommandBuilder(da);
        da.SelectCommand = cmd;
        ds = new DataSet();
        da.Fill(ds, "hasilakhir");
        bs = new BindingSource();
        bs.DataSource = ds.Tables["hasilakhir"];
        dgvKota.DataSource = bs;

        dgvKota.Columns[0].HeaderCell.Value = "Propinsi";
        dgvKota.Columns[1].HeaderCell.Value = "Jenis";
        dgvKota.Columns[2].HeaderCell.Value = "Kota";
        dgvKota.Columns[3].HeaderCell.Value = "Peningkatan";
        dgvKota.Columns[4].HeaderCell.Value = "Rata2 Naik";
        dgvKota.Columns[5].HeaderCell.Value = "Rata2 Turun";
        dgvKota.Columns[6].HeaderCell.Value = "Jum Mhs";
        dgvKota.Columns[7].HeaderCell.Value = "Jarak";
        dgvKota.Columns[8].HeaderCell.Value = "Jum SMA";
        dgvKota.Columns[9].HeaderCell.Value = "Rata2 IPK";
        dgvKota.Columns[0].Width = 140;
        dgvKota.Columns[1].Width = 53;
        dgvKota.Columns[2].Width = 140;
        dgvKota.Columns[3].Width = 77;
        dgvKota.Columns[4].Width = 43;
        dgvKota.Columns[5].Width = 43;
        dgvKota.Columns[6].Width = 40;
        dgvKota.Columns[7].Width = 40;
        dgvKota.Columns[8].Width = 40;
        dgvKota.Columns[9].Width = 43;
        chart1.Titles.RemoveAt(0);
        chart1.Series[("Tahun")].Points.Clear();
        dgvKota.Focus();
        Chart(); DaftarSMA();
    }

    private void Chart()
    {
        chart1.Titles.Add("Jumlah Mahasiswa Pertahun");
        string strSql = @"select tahun1, tahun2, tahun3, tahun4, tahun5,jumlah from
       peningkatan where kd_kota=@kd_kota";
        cmd = new SqlCommand(strSql, conn);
        cmd.Parameters.AddWithValue("@kd_kota", kd_kota);
        conn.Close();
        conn.Open();

        dr = cmd.ExecuteReader();
        if (dr.HasRows)
        {
            while (dr.Read())
            {
                chart1.Series[("Tahun")].Points.AddY(Convert.ToInt32(dr["tahun1"]));
                chart1.Series[("Tahun")].Points.AddY(Convert.ToInt32(dr["tahun2"]));
                chart1.Series[("Tahun")].Points.AddY(Convert.ToInt32(dr["tahun3"]));
                chart1.Series[("Tahun")].Points.AddY(Convert.ToInt32(dr["tahun4"]));
                chart1.Series[("Tahun")].Points.AddY(Convert.ToInt32(dr["tahun5"]));
            }
        }
    }

    private void DaftarSMA()
    {
        string strSql = @"select distinct(sma) from mahasiswa where kd_kota=@kd_kota";
        cmd = new SqlCommand(strSql, conn);
        cmd.Parameters.AddWithValue("@kd_kota", kd_kota);
        conn.Close();
        conn.Open();
```

```
        da = new SqlDataAdapter();
        cb = new SqlCommandBuilder(da);
        da.SelectCommand = cmd;
        ds = new DataSet();
        da.Fill(ds, "sma");
        bs = new BindingSource();
        bs.DataSource = ds.Tables["sma"];
        dgvSma.DataSource = bs;

        dgvSma.Columns[0].HeaderCell.Value = "Daftar SMA";
        dgvSma.Columns[0].Width = 260;
    }

    private void MaksMin()
    {
        string strSql = @"select max(peningkatan) as makspen, min(peningkatan) as
                          minpen, max(jum_mhs) as maksMhs, min(jum_mhs) as minMhs,
                          max(jum_sma) as maksSms, min(jum_sma) as minSma from resume,
                          peningkatan where resume.kd_kota = peningkatan.kd_kota";
        cmd = new SqlCommand(strSql, conn);
        conn.Close();
        conn.Open();
        dr = cmd.ExecuteReader();
        if (dr.HasRows)
        {
            while (dr.Read())
            {
                lblMinMeningkat.Text = dr["minpen"].ToString();
                lblMaksMeningkat.Text = dr["makspen"].ToString();
                lblMinMhs.Text = dr["minMhs"].ToString();
                lblMaksMhs.Text = dr["maksMhs"].ToString();
                lblMinSma.Text = dr["minSma"].ToString();
                lblMaksSma.Text = dr["maksSms"].ToString();
            }
        }
    }

    #region NumberingMode
    private void txtAwalPen_KeyPress(object sender, KeyPressEventArgs e)
    {
        if (!char.IsControl(e.KeyChar) && !char.IsDigit(e.KeyChar) && e.KeyChar != '.'
        && e.KeyChar != '-')
        {
            e.Handled = true;
        }

        // only allow one decimal point
        if (e.KeyChar == '.' && (sender as TextBox).Text.IndexOf('.') > -1)
        {
            e.Handled = true;
        }
    }

    private void txtAkhirPen_KeyPress(object sender, KeyPressEventArgs e)
    {
        if (!char.IsControl(e.KeyChar) && !char.IsDigit(e.KeyChar) && e.KeyChar != '.'
         && e.KeyChar != '-')
        {
            e.Handled = true;
        }

        // only allow one decimal point
        if (e.KeyChar == '.' && (sender as TextBox).Text.IndexOf('.') > -1)
        {
            e.Handled = true;
```

```
    }
}


private void txtJumMhsAwal_KeyPress(object sender, KeyPressEventArgs e)
{
    if (!char.IsControl(e.KeyChar) && !char.IsDigit(e.KeyChar) && e.KeyChar != '.')
    {
        e.Handled = true;
    }

     //only allow one decimal point
    if (e.KeyChar == '.' && (sender as TextBox).Text.IndexOf('.') > -1)
    {
        e.Handled = true;
    }
}

private void txtJumMhsAkhir_KeyPress(object sender, KeyPressEventArgs e)
{
    if (!char.IsControl(e.KeyChar) && !char.IsDigit(e.KeyChar) && e.KeyChar != '.')
    {
        e.Handled = true;
    }

     //only allow one decimal point
    if (e.KeyChar == '.' && (sender as TextBox).Text.IndexOf('.') > -1)
    {
        e.Handled = true;
    }
}

private void txtJumSmaAwal_KeyPress(object sender, KeyPressEventArgs e)
{
    if (!char.IsControl(e.KeyChar) && !char.IsDigit(e.KeyChar) && e.KeyChar != '.')
    {
        e.Handled = true;
    }

     //only allow one decimal point
    if (e.KeyChar == '.' && (sender as TextBox).Text.IndexOf('.') > -1)
    {
        e.Handled = true;
    }
}

private void txtJumSmaAkhir_KeyPress(object sender, KeyPressEventArgs e)
{
    if (!char.IsControl(e.KeyChar) && !char.IsDigit(e.KeyChar) && e.KeyChar != '.')
    {
        e.Handled = true;
    }

     //only allow one decimal point
    if (e.KeyChar == '.' && (sender as TextBox).Text.IndexOf('.') > -1)
    {
        e.Handled = true;
    }
}

private void txtJumSaring_KeyPress(object sender, KeyPressEventArgs e)
{
    if (!char.IsControl(e.KeyChar) && !char.IsDigit(e.KeyChar) && e.KeyChar != '.')
    {
        e.Handled = true;
    }
```

```
     //only allow one decimal point
    if (e.KeyChar == '.' && (sender as TextBox).Text.IndexOf('.') > -1)
    {
        e.Handled = true;
    }
}
#endregion

#region ValidationMode
private void txtAwalPen_Leave(object sender, EventArgs e)
{
    if (txtAwalPen.Text != "" && Convert.ToDouble(txtAwalPen.Text.Trim()) <
    Convert.ToDouble(lblMinMeningkat.Text.Trim()))
    {
        MessageBox.Show("Tidak Boleh Lebih Kecil Dari Minimal", "Perhatian",
        MessageBoxButtons.OK, MessageBoxIcon.Stop,
        MessageBoxDefaultButton.Button1);
        txtAwalPen.Text = "";
        txtAwalPen.Focus();
    }
}

private void txtJumMhsAwal_Leave(object sender, EventArgs e)
{
    if (txtJumMhsAwal.Text != "" && Convert.ToInt32(txtJumMhsAwal.Text.Trim()) <
    Convert.ToInt32(lblMinMhs.Text.Trim()))
    {
        MessageBox.Show("Tidak Boleh Lebih Kecil Dari Minimal", "Perhatian",
        MessageBoxButtons.OK, MessageBoxIcon.Stop,
        MessageBoxDefaultButton.Button1);
        txtJumMhsAwal.Text = "";
        txtJumMhsAwal.Focus();
    }
}

private void txtJumSmaAwal_Leave(object sender, EventArgs e)
{
    if (txtJumSmaAwal.Text != "" && Convert.ToInt32(txtJumSmaAwal.Text.Trim()) <
    Convert.ToInt32(lblMinSma.Text.Trim()))
    {
        MessageBox.Show("Tidak Boleh Lebih Kecil Dari Minimal", "Perhatian",
        MessageBoxButtons.OK, MessageBoxIcon.Stop,
        MessageBoxDefaultButton.Button1);
        txtJumMhsAwal.Text = "";
        txtJumMhsAwal.Focus();
    }
}
#endregion

private void btnBobot_Click(object sender, EventArgs e)
{
    SetupBobot setupBbt = new SetupBobot();
    setupBbt.ShowDialog();
    setupBbt.Dispose();
    this.Activate();
    this.Show();
}

private void btnKeluar_Click(object sender, EventArgs e)
{
    this.Close();
}

private void BtnSaring_Click(object sender, EventArgs e)
{
```

```
    string a = txtAwalPen.Text; string b = txtAkhirPen.Text;
    string c = txtJumMhsAwal.Text; string d = txtJumSmaAkhir.Text;
    var i = cmbJarak.SelectedItem;
    string f = txtJumSmaAwal.Text; string g = txtJumSmaAkhir.Text;
    var h = cmbRrIpk.SelectedItem;

    cmbJarakStr = cmbJarak.SelectedItem.ToString();
    cmbRerataIpkStr = cmbRrIpk.SelectedItem.ToString();
    cmbPerbandinganStr = cmbPerbandingan.SelectedItem.ToString();
    cmbTahunStr = cmbTahun.SelectedItem.ToString();

    if (a == "" || b == "" || c == "" || d == "" ||
        i == null || f == "" || g == "" || h == null)
    {
        MessageBox.Show("Lengkapi semua data", "Perhatian", MessageBoxButtons.OK,
        MessageBoxIcon.Stop, MessageBoxDefaultButton.Button1);
    }
    else
    {
        if (backgroundWorker1.IsBusy != true)
        {
            // create a new instance of the alert form
            alert = new AlertForm();
            // event handler for the Cancel button in AlertForm
            alert.Canceled += new EventHandler<EventArgs>(buttonCancel_Click);
            //this.Opacity = Convert.ToInt32(1);
            alert.Show();

            backgroundWorker1.RunWorkerAsync();
            // Start the asynchronous operation.
        }
    }
}

private void dgvKota_CellMouseClick(object sender, DataGridViewCellMouseEventArgs
 e)
{
    if (e.RowIndex == 0 || e.RowIndex > 0)
    {
        nmx = dgvKota.Rows[e.RowIndex].Cells[2].FormattedValue.ToString();
        je = dgvKota.Rows[e.RowIndex].Cells[1].FormattedValue.ToString();

        string strSql = @"select kd_kota from kota where nm_kota=@nm_kota and
        jenis=@jenis";
        cmd = new SqlCommand(strSql, conn);
        cmd.Parameters.AddWithValue("@nm_kota", nmx);
        cmd.Parameters.AddWithValue("@jenis", je);
        conn.Close();
        conn.Open();
        dr = cmd.ExecuteReader();
        if (dr.HasRows)
        {
            while (dr.Read())
            {
                kd_kota = dr["kd_kota"].ToString();
            }
        }
    }

    chart1.Titles.RemoveAt(0);
    chart1.Series[("Tahun")].Points.Clear();

    Chart();
    DaftarSMA();
}
```

```
private void dgvKota_RowEnter(object sender, DataGridViewCellEventArgs e)
{
    if (e.RowIndex == 0 || e.RowIndex > 0)
    {
        nmx = dgvKota.Rows[e.RowIndex].Cells[2].FormattedValue.ToString();
        je = dgvKota.Rows[e.RowIndex].Cells[1].FormattedValue.ToString();

        string strSql = @"select kd_kota from kota where nm_kota=@nm_kota and
        jenis=@jenis";
        cmd = new SqlCommand(strSql, conn);
        cmd.Parameters.AddWithValue("@nm_kota", nmx);
        cmd.Parameters.AddWithValue("@jenis", je);
        conn.Close();
        conn.Open();
        dr = cmd.ExecuteReader();
        if (dr.HasRows)
        {
            while (dr.Read())
            {
                kd_kota = dr["kd_kota"].ToString();
            }
        }
    }

    chart1.Titles.RemoveAt(0);
    chart1.Series[("Tahun")].Points.Clear();

    Chart();
    DaftarSMA();
}

private void buttonCancel_Click(object sender, EventArgs e)
{
    if (backgroundWorker1.WorkerSupportsCancellation == true)
    {
        // Cancel the asynchronous operation.
        backgroundWorker1.CancelAsync();
        // Close the AlertForm
        alert.Close();
        this.Close();
    }
}

private void backgroundWorker1_DoWork(object sender, DoWorkEventArgs e)
{
    BackgroundWorker worker = sender as BackgroundWorker;

    for (int i = 1; i <= 10; i++)
    {
        if (worker.CancellationPending == true)
        {
            e.Cancel = true;
            break;
        }
        else
        {
            if (i == 1)
            { HapusHasil(); }
            else if (i == 2)
            { CopyKodeKota(); }
            else if (i == 3)
            { Peningkatan(); }
            else if (i == 4)
            { JumlahMhs(); }
            else if (i == 5)
            { JarakKota(); }
```

```
                else if (i == 6)
                { JumlahSMA(); }
                else if (i == 7)
                { RerataIPK();}
                else if (i == 8)
                { IsiLstHasil(); }
                else { JumlahNilai(); }

                // Perform a time consuming operation and report progress.
                worker.ReportProgress(i * 10);
                System.Threading.Thread.Sleep(500);
            }
        }
    }

    private void backgroundWorker1_ProgressChanged(object sender,
     ProgressChangedEventArgs e)
    {
        //Show the progress in main form (GUI)
        // Pass the progress to AlertForm label and progressbar
        alert.Message = "Sedang memproses, ... " + e.ProgressPercentage.ToString() +
"%";
        alert.ProgressValue = e.ProgressPercentage;
    }

    private void backgroundWorker1_RunWorkerCompleted(object sender,
     RunWorkerCompletedEventArgs e)
    {
        alert.Close();
        IsiDgvKota();
        tabControl1.SelectedTab = tabPage2;
    }

}
}
```