

Materi:

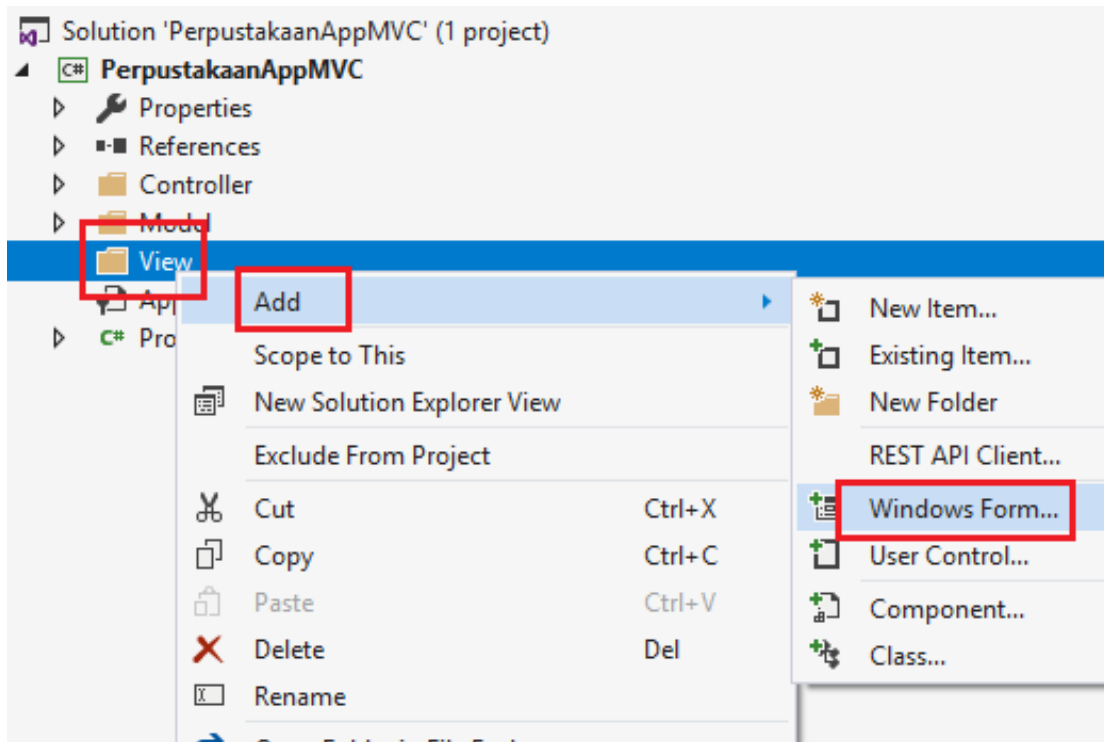
- ✓ Basic Operasi CRUD Menggunakan MVC Pattern Part #2 (View)

Pada praktikum kali ini, kita masih melanjutkan pembuatan aplikasi menggunakan MVC Pattern. Di pertemuan sebelumnya kita sudah menyelesaikan penulisan kode untuk komponen Model dan Controller, sedangkan pada praktikum kali ini kita akan melanjutkan penulisan kode untuk komponen View.

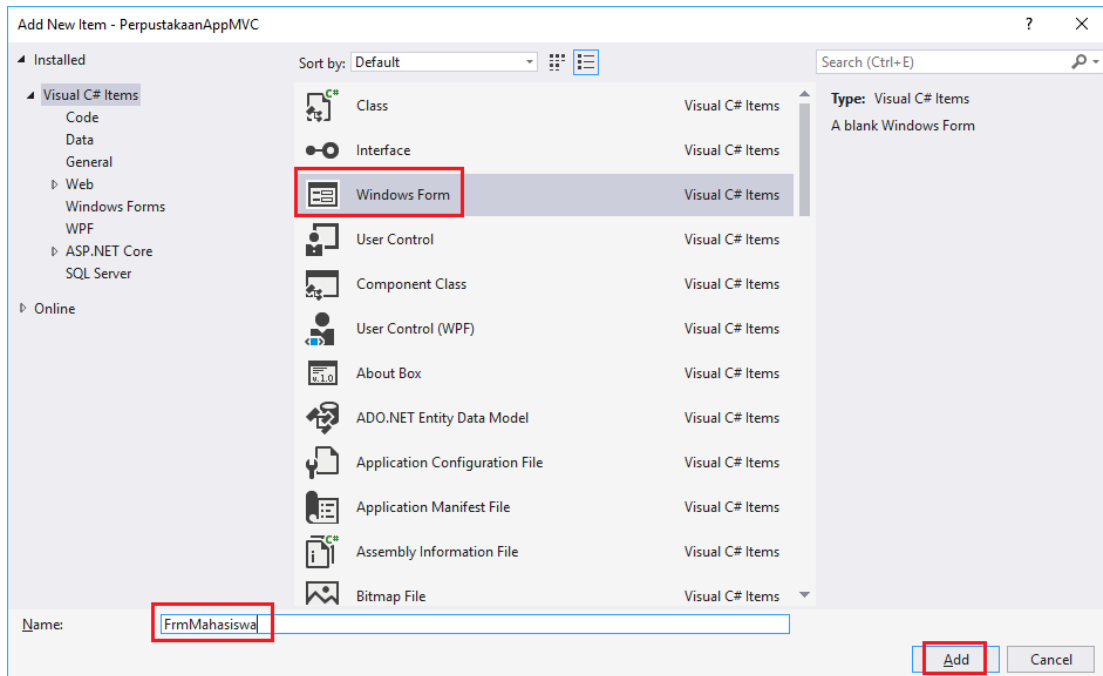
Komponen *View* bertugas untuk mengatur tampilan. Selain itu View juga bertugas untuk menerima dan mempresentasikan data kepada pengguna.

Latihan 11.1 – Menambahkan Form Mahasiswa

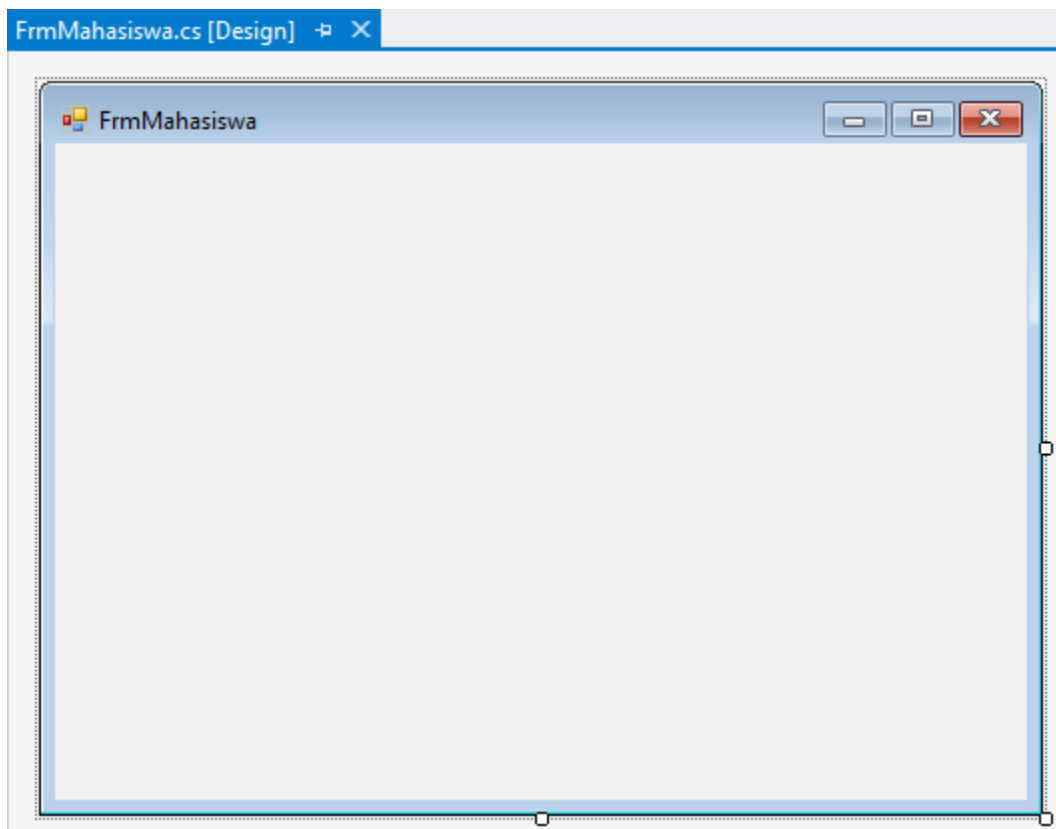
1. Klik kanan folder View -> Add -> Windows Form



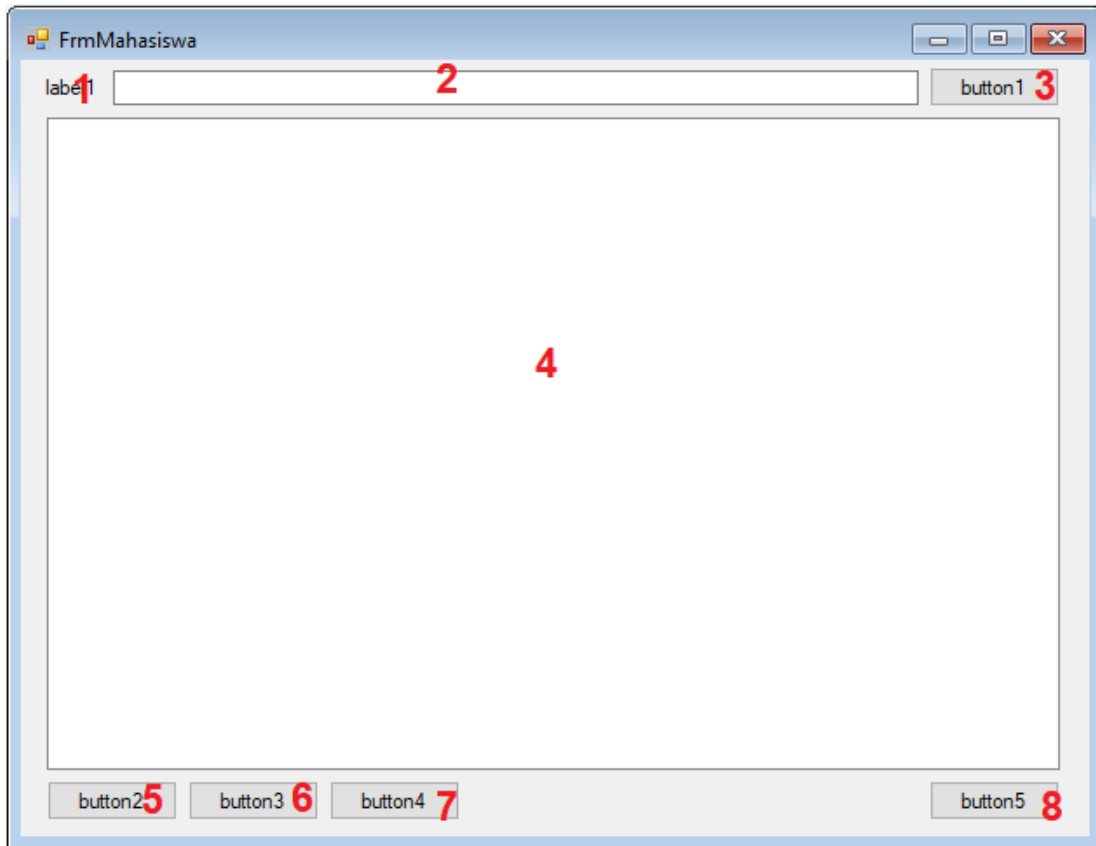
Kemudian pilih Windows Form, untuk isian Name diisi dengan **FrmMahasiswa**.



Setelah itu akan tampil form mahasiswa.



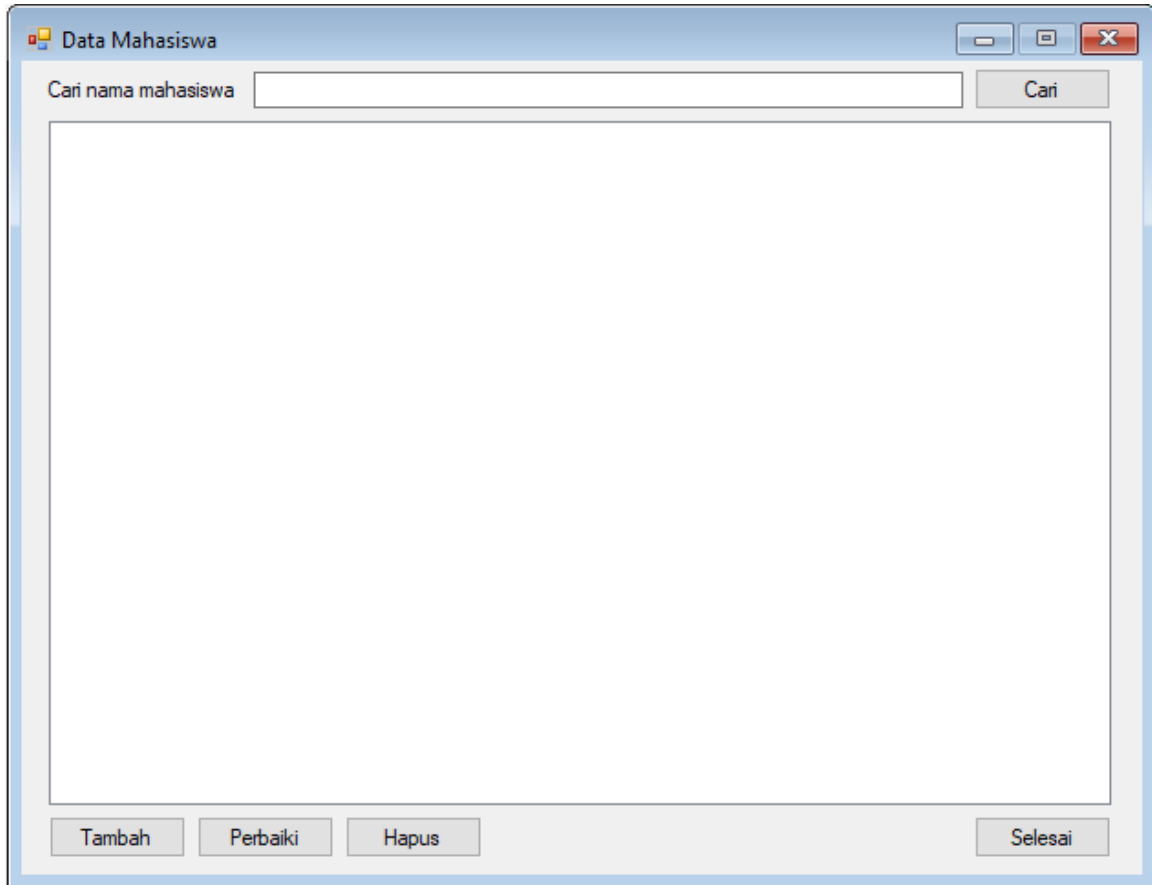
2. Kemudian desain tampilan formnya seperti gambar di bawah ini:



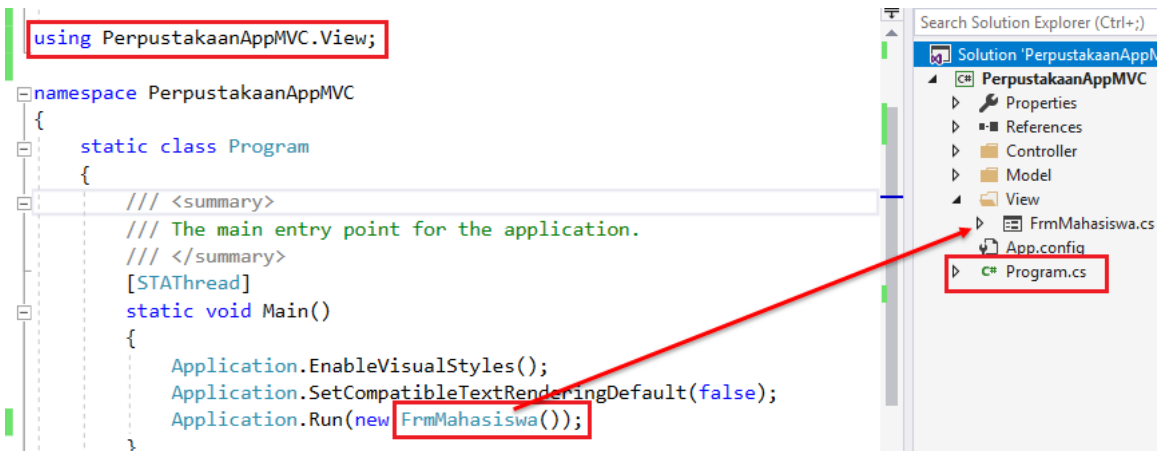
3. Kemudian atur properties masing-masing komponen melalui properties window dengan aturan sesuai tabel berikut:

No	Komponen	Property	Nilai/Value
-	Form	Text StartPosition MaximizeBox	Data Mahasiswa Center False
1	Label	Text	Cari nama mahasiswa
2	TextBox	Name	txtNama
3	Button	Name Text	btnCari Cari
4	ListView	Name	lvwMahasiswa
5	Button	Name Text	btnTambah Tambah
6	Button	Name Text	btnPerbaiki Perbaiki
7	Button	Name Text	btnHapus Hapus
8	Button	Name Text	btnSelesai Selesai

4. Setelah pengaturan properties akan didapat tampilan seperti berikut:

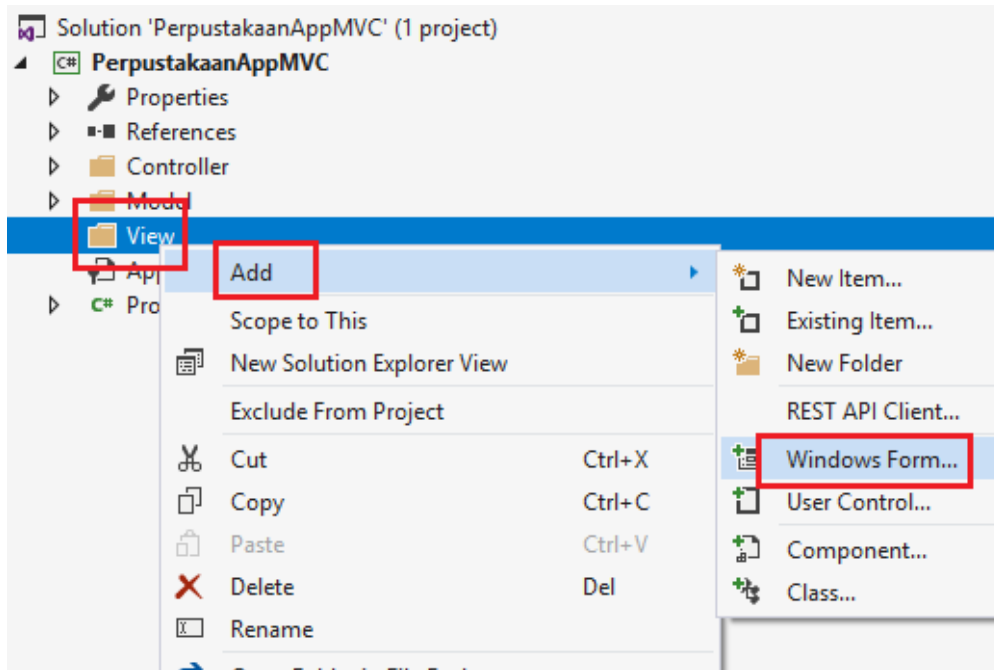


5. Kemudian arahkan entry point class Program ke Form Mahasiswa

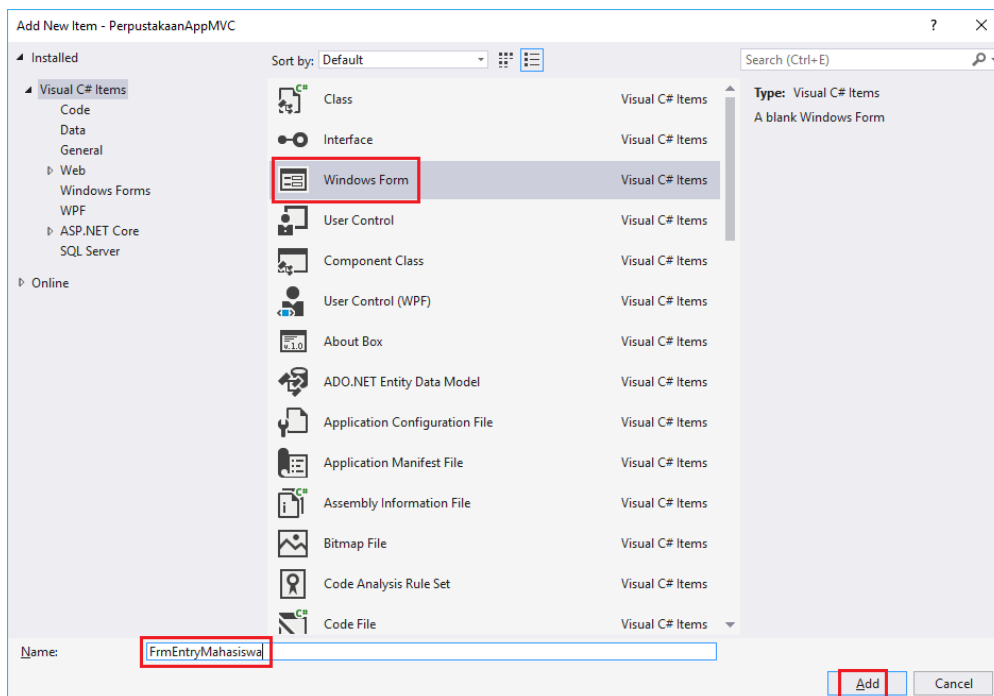


Latihan 11.2 – Menambahkan Form Entry Mahasiswa

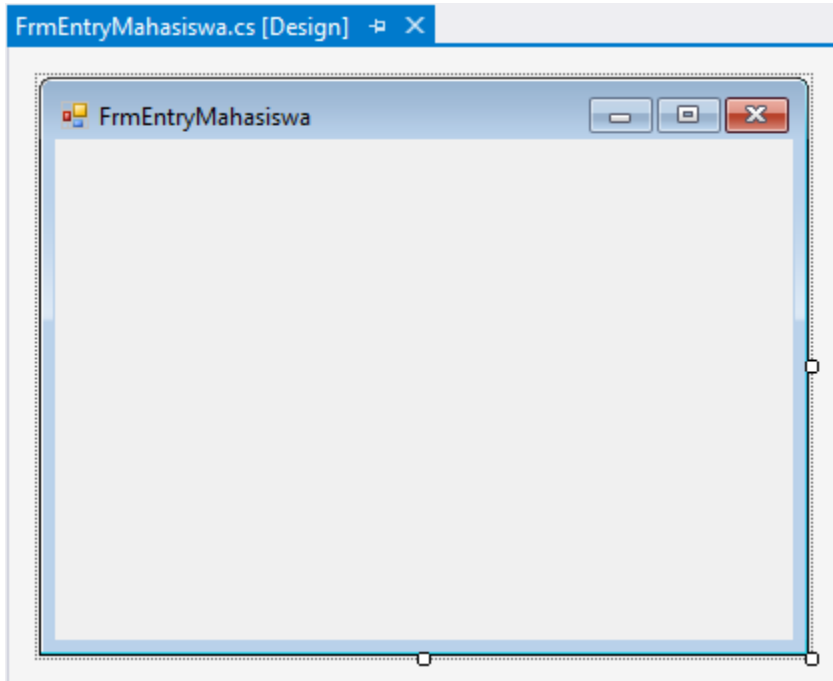
1. Klik kanan folder View -> Add -> Windows Form



Kemudian pilih Windows Form, untuk isian Name diisi dengan **FrmEntryMahasiswa**.



Setelah itu akan tampil form entry mahasiswa.



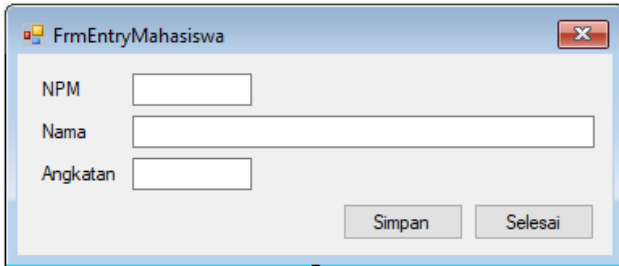
2. Kemudian desain tampilan formnya seperti gambar di bawah ini:



3. Kemudian atur properties masing-masing komponen melalui properties window dengan aturan sesuai tabel berikut:

No	Komponen	Property	Nilai/Value
-	Form	FormBorderStyle StartPosition MinimizeBox MaximizeBox ShowInTaskbar	FixedSingle CenterScreen False False False
1	TextBox	Name	txtNpm
2	TextBox	Name	txtNama
3	TextBox	Name	txtAngkatan
4	Button	Name Text	btnSimpan Simpan
7	Button	Name Text	btnSelesai Selesai

4. Setelah pengaturan properties akan didapat tampilan sebagai berikut:



Latihan 11.3 – Melengkapi Kode untuk Form Entry Mahasiswa

1. Setelah menambahkan form entry mahasiswa, langkah berikutnya kita akan mendeklarasikan delegate dan event. Ada 2 event yang akan kita buat yaitu event *OnCreate* dan *OnUpdate*. Kedua event ini dipanggil pada saat user menginputkan data baru (*OnCreate*) dan ketika user mengupdate data (*OnUpdate*). Sebelum mendeklarasikan *event* terlebih dulu kita harus mendeklarasikan tipe dari *event* tersebut dengan menggunakan *delegate*. Aktifkan editor code form entry mahasiswa, kemudian lengkapi kode untuk mendeklarasikan delegate dan event seperti berikut:

```
// deklarasi tipe data untuk event OnCreate dan OnUpdate
public delegate void CreateUpdateEventHandler(Mahasiswa mhs);

public partial class FrmEntryMahasiswa : Form
{
    // deklarasi event ketika terjadi proses input data baru
    public event CreateUpdateEventHandler OnCreate;

    // deklarasi event ketika terjadi proses update data
    public event CreateUpdateEventHandler OnUpdate;

    // deklarasi objek controller
    private MahasiswaController controller;

    // deklarasi field untuk menyimpan status entry data (input baru atau
    update)
    private bool isNewData = true;

    // deklarasi field untuk meyimpan objek mahasiswa
    private Mahasiswa mhs;

    // constructor default
    public FrmEntryMahasiswa()
    {
        InitializeComponent();
    }
}
```

Pada kode di atas, selain mendeklarasikan *delegate* dan *event*, kita juga mendeklarasikan tiga buah field yaitu *controller*, *isNewData* dan *mhs*. Field *controller* digunakan untuk menyimpan objek dari class MahasiswaController, *isNewData* berfungsi untuk menyimpan status entri data apakah sedang menginput data baru (create) atau mengupdate data yang sudah ada (update). Sedangkan field *mhs* berfungsi untuk menyimpan objek dari class Mahasiswa.

Selain menambahkan kode di atas, kita juga perlu mendaftarkan namespace *PerpustakaanAppMVC.Model.Entity* dan *PerpustakaanAppMVC.Controller*, agar form entry mahasiswa bisa mengakses class Mahasiswa dan Controller.

```
using System.Windows.Forms;

using PerpustakaanAppMVC.Model.Entity;
using PerpustakaanAppMVC.Controller;

namespace PerpustakaanAppMVC.View
{
    // deklarasi tipe data untuk event OnCreate dan OnUpdate
    public delegate void CreateUpdateEventHandler(Mahasiswa mhs);

    public partial class FrmEntryMahasiswa : Form
    {
        // deklarasi event ketika terjadi proses input data baru
        public event CreateUpdateEventHandler OnCreate;
    }
}
```

2. Selanjutnya kita akan menambahkan dua buah constructor lagi yang digunakan untuk keperluan inisialisasi data ketika terjadi proses input data baru (create) atau edit data (update). Masih di editor code yang sama, kemudian tambahkan dua constructor berikut setelah constructor default.


```
// constructor untuk inisialisasi data ketika entri data baru
public FrmEntryMahasiswa(string title, MahasiswaController controller)
    : this()
{
    // ganti text/judul form
    this.Text = title;
    this.controller = controller;
}

// constructor untuk inisialisasi data ketika mengedit data
public FrmEntryMahasiswa(string title, Mahasiswa obj, MahasiswaController
controller)
    : this()
{
    // ganti text/judul form
    this.Text = title;
    this.controller = controller;

    isNewData = false; // set status edit data
    mhs = obj; // set objek mhs yang akan diedit

    // untuk edit data, tampilkan data lama
    txtNpm.Text = mhs.Npm;
    txtNama.Text = mhs>Nama;
    txtAngkatan.Text = mhs.Angkatan;
}
```

Constructor adalah method khusus yang mempunyai nama yang sama dengan nama classnya dan dipanggil secara otomatis ketika sebuah objek dibuat. Jadi ketika objek FrmEntryMahasiswa dibuat, maka secara otomatis semua kode yang ada di blok constructor tersebut akan dieksekusi tergantung constructor mana yang dipanggil.

3. Langkah berikutnya kita akan melengkapi kode untuk tombol Simpan, caranya adalah dengan mengaktifkan event handler *btnSimpan_Click*.

```
private void btnSimpan_Click(object sender, EventArgs e)
{
    ...
}
```

Kemudian lengkapi kodenya seperti berikut:

```
private void btnSimpan_Click(object sender, EventArgs e)
{
    // jika data baru, inisialisasi objek mahasiswa
    if (isNewData) mhs = new Mahasiswa();

    // set nilai property objek mahasiswa yg diambil dari TextBox
    mhs.Npm = txtNpm.Text;
    mhs>Nama = txtNama.Text;
    mhs.Angkatan = txtAngkatan.Text;

    int result = 0;

    if (isNewData) // tambah data baru, panggil method Create
    {
        // panggil operasi CRUD
        result = controller.Create(mhs);

        if (result > 0) // tambah data berhasil
        {
            OnCreate(mhs); // panggil event OnCreate

            // reset form input, utk persiapan input data berikutnya
            txtNpm.Clear();
            txtNama.Clear();
            txtAngkatan.Clear();

            txtNpm.Focus();
        }
    }
    else // edit data, panggil method Update
    {
        // panggil operasi CRUD
        result = controller.Update(mhs);

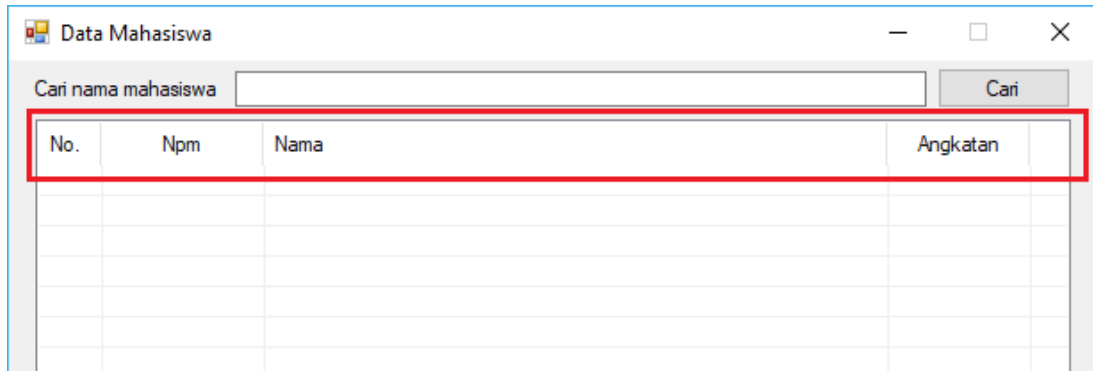
        if (result > 0)
        {
            OnUpdate(mhs); // panggil event OnUpdate
            this.Close();
        }
    }
}
```

4. Terakhir lengkapi juga kode untuk tombol Selesai seperti berikut:

```
private void btnSelesai_Click(object sender, EventArgs e)
{
    // tutup form entry data mahasiswa
    this.Close();
}
```

Latihan 11.4 – Melengkapi Kode untuk Form Mahasiswa

1. Hal pertama yang kita lakukan adalah mengatur property ListView agar mempunyai tampilan seperti berikut:



Caranya dengan mengaktifkan editor code, kemudian tambahkan method InisialisasiListView berikut di bawah constructor.

```
// atur kolom listview
private void InisialisasiListView()
{
    lvwMahasiswa.View = System.Windows.Forms.View.Details;
    lvwMahasiswa.FullRowSelect = true;
    lvwMahasiswa.GridLines = true;

    lvwMahasiswa.Columns.Add("No.", 35, HorizontalAlignment.Center);
    lvwMahasiswa.Columns.Add("Npm", 91, HorizontalAlignment.Center);
    lvwMahasiswa.Columns.Add("Nama", 350, HorizontalAlignment.Left);
    lvwMahasiswa.Columns.Add("Angkatan", 80, HorizontalAlignment.Center);
}
```

Untuk petunjuknya lihat gambar berikut:

```
public partial class FrmMahasiswa : Form
{
    // constructor
    public FrmMahasiswa()
    {
        InitializeComponent();
    }

    // atur kolom listview
    private void InisialisasiListView()
    {
        lvwMahasiswa.View = View.Details;
        lvwMahasiswa.FullRowSelect = true;
        lvwMahasiswa.GridLines = true;
    }
}
```

Setelah itu panggil method InisialisasiListView dari dalam constructor.

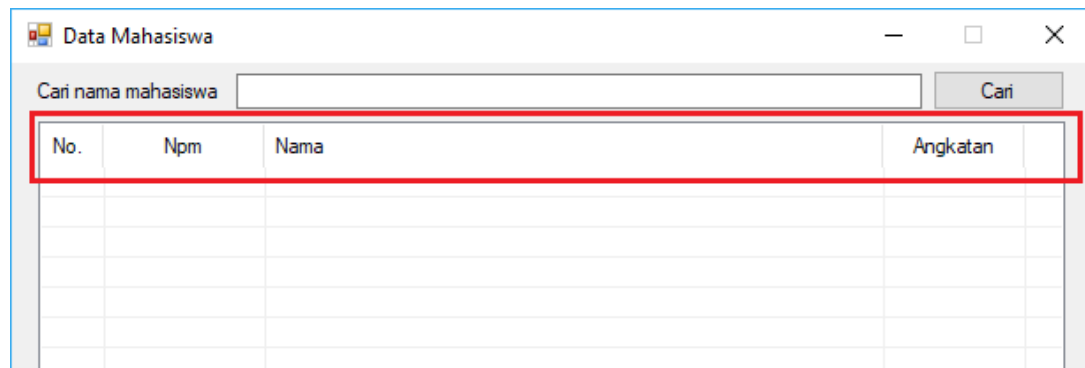
```
namespace PerpustakaanAppMVC.View
{
    public partial class FrmMahasiswa : Form
    {
        // constructor
        public FrmMahasiswa()
        {
            InitializeComponent();
            InisialisasiListView();
        }

        // atur kolom listview
        private void InisialisasiListView()
        {

```

Constructor adalah method khusus yang mempunyai nama yang sama dengan nama classnya dan dipanggil secara otomatis ketika sebuah objek dibuat. Jadi ketika objek FrmMahasiswa dibuat, secara otomatis method InisialisasiListView akan dipanggil untuk memformat tampilan ListView.

2. Kemudian jalankan aplikasi untuk melihat hasil sementara dengan menekan tombol F5 (Start Debugging)



3. Kembali lagi ke editor kode, kemudian tambahkan kode berikut untuk mendeklarasikan dua buah field. Field pertama dengan tipe collection yang digunakan untuk menyimpan kumpulan objek dari class Mahasiswa, dan field yang kedua dengan tipe controller untuk menjalankan operasi CRUD.

```
using System.Windows.Forms;

using PerpustakaanAppMVC.Model.Entity;
using PerpustakaanAppMVC.Controller;

namespace PerpustakaanAppMVC.View
{
    public partial class FrmMahasiswa : Form
    {
        // deklarasi objek collection untuk menampung objek mahasiswa
        private List<Mahasiswa> listOfMahasiswa = new List<Mahasiswa>();

        // deklarasi objek controller
        private MahasiswaController controller;

        // constructor
        public FrmMahasiswa()
        {
            InitializeComponent();

            // membuat objek controller
            controller = new MahasiswaController();

            InisialisasiListView();
        }
    }
}
```

4. Langkah berikutnya kita akan menambahkan sebuah method dengan nama *LoadDataMahasiswa* yang digunakan untuk menampilkan data mahasiswa ke ListView. Method ini ditambahkan setelah method *InisialisasiListView*, berikut kode lengkapnya:

```
// method untuk menampilkan semua data mahasiswa
private void LoadDataMahasiswa()
{
    // kosongkan listview
    lvwMahasiswa.Items.Clear();

    // panggil method ReadAll dan tampung datanya ke dalam collection
    listOfMahasiswa = controller.ReadAll();

    // ekstrak objek mhs dari collection
    foreach (var mhs in listOfMahasiswa)
    {
        var noUrut = lvwMahasiswa.Items.Count + 1;

        var item = new ListViewItem(noUrut.ToString());
        item.SubItems.Add(mhs.Npm);
        item.SubItems.Add(mhs>Nama);
        item.SubItems.Add(mhs.Angkatan);

        // tampilkan data mhs ke listview
        lvwMahasiswa.Items.Add(item);
    }
}
```

Setelah itu panggil juga method LoadDataMahasiswa dari dalam constructor.

```
public partial class FrmMahasiswa : Form
{
    // deklarasi objek collection untuk menampung objek mahasiswa
    private List<Mahasiswa> listOfMahasiswa = new List<Mahasiswa>();

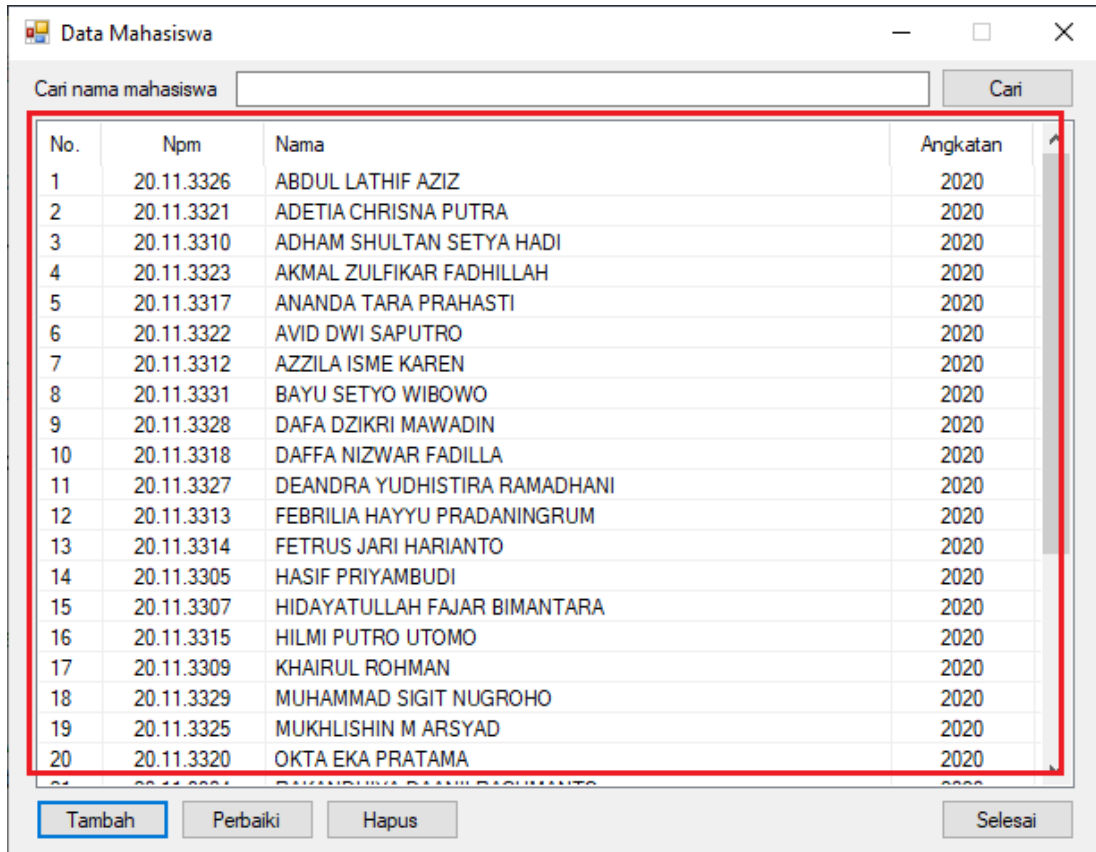
    // deklarasi objek controller
    private MahasiswaController controller;

    // constructor
    public FrmMahasiswa()
    {
        InitializeComponent();

        // membuat objek controller
        controller = new MahasiswaController();

        InisialisasiListView();
        LoadDataMahasiswa();
    }
}
```

5. Kemudian jalankan aplikasi untuk melihat hasil sementara dengan menekan tombol F5 (Start Debugging)



No.	Npm	Nama	Angkatan
1	20.11.3326	ABDUL LATHIF AZIZ	2020
2	20.11.3321	ADETIA CHRISNA PUTRA	2020
3	20.11.3310	ADHAM SHULTAN SETYA HADI	2020
4	20.11.3323	AKMAL ZULFIKAR FADHILLAH	2020
5	20.11.3317	ANANDA TARA PRAHASTI	2020
6	20.11.3322	AVID DWI SAPUTRO	2020
7	20.11.3312	AZZILA ISME KAREN	2020
8	20.11.3331	BAYU SETYO WIBOWO	2020
9	20.11.3328	DAFA DZIKRI MAWADIN	2020
10	20.11.3318	DAFFA NIZWAR FADILLA	2020
11	20.11.3327	DEANDRA YUDHISTIRA RAMADHANI	2020
12	20.11.3313	FEBRILIA HAYYU PRADANINGRUM	2020
13	20.11.3314	FETRUS JARI HARIANTO	2020
14	20.11.3305	HASIF PRIYAMBUDI	2020
15	20.11.3307	HIDAYATULLAH FAJAR BIMANTARA	2020
16	20.11.3315	HILMI PUTRO UTOMO	2020
17	20.11.3309	KHAIRUL ROHMAN	2020
18	20.11.3329	MUHAMMAD SIGIT NUGROHO	2020
19	20.11.3325	MUKHLISHIN M ARSYAD	2020
20	20.11.3320	OKTA EKA PRATAMA	2020

6. Langkah berikutnya kita akan menambahkan dua buah event handler yang berfungsi untuk merespon event *OnCreate* dan *OnUpdate* ketika dipanggil dari form entry mahasiswa.

```
// method event handler untuk merespon event OnCreate,
private void OnCreateEventHandler(Mahasiswa mhs)
{
    // tambahkan objek mhs yang baru ke dalam collection
    listOfMahasiswa.Add(mhs);

    int noUrut = lvwMahasiswa.Items.Count + 1;

    // tampilkan data mhs yg baru ke list view
    ListViewItem item = new ListViewItem(noUrut.ToString());
    item.SubItems.Add(mhs.Npm);
    item.SubItems.Add(mhs>Nama);
    item.SubItems.Add(mhs.Angkatan);

    lvwMahasiswa.Items.Add(item);
}

// method event handler untuk merespon event OnUpdate,
private void OnUpdateEventHandler(Mahasiswa mhs)
{
    // ambil index data mhs yang edit
    int index = lvwMahasiswa.SelectedIndices[0];

    // update informasi mhs di listview
    ListViewItem itemRow = lvwMahasiswa.Items[index];
    itemRow.SubItems[1].Text = mhs.Npm;
    itemRow.SubItems[2].Text = mhs>Nama;
    itemRow.SubItems[3].Text = mhs.Angkatan;
}
```

Kedua method di atas ditambahkan setelah method LoadDataMahasiswa

- Setelah menambahkan method event handler *OnCreate* dan *OnUpdate*, langkah berikutnya adalah melengkapi kode untuk tombol Tambah. Adapun kodenya seperti berikut:

```
private void btnTambah_Click(object sender, EventArgs e)
{
    // buat objek form entry data mahasiswa
    FrmEntryMahasiswa frmEntry = new FrmEntryMahasiswa("Tambah Data
Mahasiswa", controller);

    // mendaftarkan method event handler untuk merespon event OnCreate
    frmEntry.OnCreate += OnCreateEventHandler;

    // tampilkan form entry mahasiswa
    frmEntry.ShowDialog();
}
```


8. Kemudian lengkapi juga kode untuk tombol Perbaiki seperti berikut:

```
private void btnPerbaiki_Click(object sender, EventArgs e)
{
    if (lvwMahasiswa.SelectedItems.Count > 0)
    {
        // ambil objek mhs yang mau diedit dari collection
        Mahasiswa mhs = listOfMahasiswa[lvwMahasiswa.SelectedIndices[0]];

        // buat objek form entry data mahasiswa
        FrmEntryMahasiswa frmEntry = new FrmEntryMahasiswa("Edit Data
Mahasiswa", mhs, controller);

        // mendaftarkan method event handler untuk merespon event OnUpdate
        frmEntry.OnUpdate += OnUpdateEventHandler;

        // tampilkan form entry mahasiswa
        frmEntry.ShowDialog();
    }
    else // data belum dipilih
    {
        MessageBox.Show("Data belum dipilih", "Peringatan",
        MessageBoxButtons.OK,
        MessageBoxIcon.Exclamation);
    }
}
```

9. Terakhir lengkapi juga kode untuk tombol Hapus dan Selesai.

```
private void btnHapus_Click(object sender, EventArgs e)
{
    if (lvwMahasiswa.SelectedItems.Count > 0)
    {
        var konfirmasi = MessageBox.Show("Apakah data mahasiswa ingin
dihapus?", "Konfirmasi",
        MessageBoxButtons.YesNo, MessageBoxIcon.Exclamation);

        if (konfirmasi == DialogResult.Yes)
        {
            // ambil objek mhs yang mau dihapus dari collection
            Mahasiswa mhs =
listOfMahasiswa[lvwMahasiswa.SelectedIndices[0]];

            // panggil operasi CRUD
            var result = controller.Delete(mhs);
            if (result > 0) LoadDataMahasiswa();
        }
    }
    else // data belum dipilih
    {
        MessageBox.Show("Data mahasiswa belum dipilih !!!", "Peringatan",
        MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
    }
}
```

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

10. Setelah menyelesaikan kode program, jalankan aplikasi dengan menekan tombol F5.

The screenshot shows a Windows application titled "Data Mahasiswa". It features a search bar at the top with the placeholder text "Cari nama mahasiswa" and a "Cari" button. Below the search bar is a table with four columns: "No.", "Npm", "Nama", and "Angkatan". The table contains eight rows of student data. At the bottom of the window, there are four buttons: "Tambah", "Perbaiki", "Hapus", and "Selesai".

No.	Npm	Nama	Angkatan
1	20.11.3326	ABDUL LATHIF AZIZ	2020
2	20.11.3321	ADETIA CHRISNA PUTRA	2020
3	20.11.3310	ADHAM SHULTAN SETYA HADI	2020
4	20.11.3323	AKMAL ZULFIKAR FADHILLAH	2020
5	20.11.3317	ANANDA TARA PRAHASTI	2020
6	20.11.3322	AVID DWI SAPUTRO	2020
7	20.11.3312	AZZILA ISME KAREN	2020
8	20.11.3331	BAYU SETYO WIBOWO	2020

Setelah itu Anda coba semua operasi CRUD yang ada, seperti tambah, perbaiki dan hapus.

Tugas 11.1:

- ✓ Lengkapi kode untuk tombol Cari

This screenshot shows the same "Data Mahasiswa" application window, but with the search bar containing the text "putra" and the "Cari" button highlighted with a red box. The table below shows the results of the search, with three rows highlighted by a red box. The first row is "ADETIA CHRISNA PUTRA" with NPM 20.11.3321. The second row is "RIZKY BINTANG SAPUTRA" with NPM 20.11.3330. The third row is "ROBBY AGIL SAPUTRA" with NPM 20.11.3319.

No.	Npm	Nama	Angkatan
1	20.11.3321	ADETIA CHRISNA PUTRA	2020
2	20.11.3330	RIZKY BINTANG SAPUTRA	2020
3	20.11.3319	ROBBY AGIL SAPUTRA	2020

Selesai ☺