1. **TUJUAN**
2. Memudahkan pengguna untuk mengkonversi suhu
3. Melihat kondisi cuaca di suatu tempat
4. Menyimpan hasil kedalam file \*.txt
5. **PENDAHULUAN**

Untuk mengkonversi suhu, biasanya perlu menghitung manual, apalagi jika angka tersebut adalah angka yang rumit dan berdesimal (float). Tentunya akan mudah jika menggunakan alat hitung seperti kalkulator., tetapi tetap saja perlu menuliskan rumus atau perhitungannya. Dengan program konversi suhu / temperature Celcius, Reamur, Fahrenheit, Kelvin, dan temperature lain yang titik atas dan titik bawahnya bernilai sembarang, maka akan memudahkan pengguna untuk mengkonversi suhu tersebut hanya dengan inputan suhu awal.

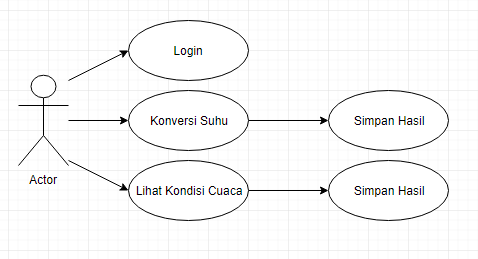
Jika seseorang ingin bepergian keluar untuk bekerja, sekolah, atau rekreasi, tentu alangkah baiknya melihat cuaca di kota tujuan. Dengan API yang disediakan *openweathermap*, kita bisa melihat cuaca di kota yang ada di dalam list. Dan apabila sesorang ingin menyimpan kondisi cuaca di kota tersebut, sesorang biasanya akan menscreenshot atau menulis ulang hasilnya. Dengan adanya fitur save result, akan memudahkan pengguna secara langsung untuk menyimpan hasil tersebut. Aplikasi dibuat dengan bahasa pemrograman C# berbasis Windows Form .NET Framework dibantu Tool Microsoft Visual Studio.

Tahapan pembuatan aplikasi:

1. Form desain / User Interface
2. Membuat Usecase dari aplikasi dan Activity Diagram
3. Membuat database user
4. Menerapkan Activity Diagram kedalam code:
   1. Login dan Register dengan sqlconnection
   2. Konversi Suhu
   3. Terhubung dengan API openweathermap dan save result
5. Menguji aplikasi
6. **PEMBAHASAN**
   1. **Usecase dan Activity Diagram**

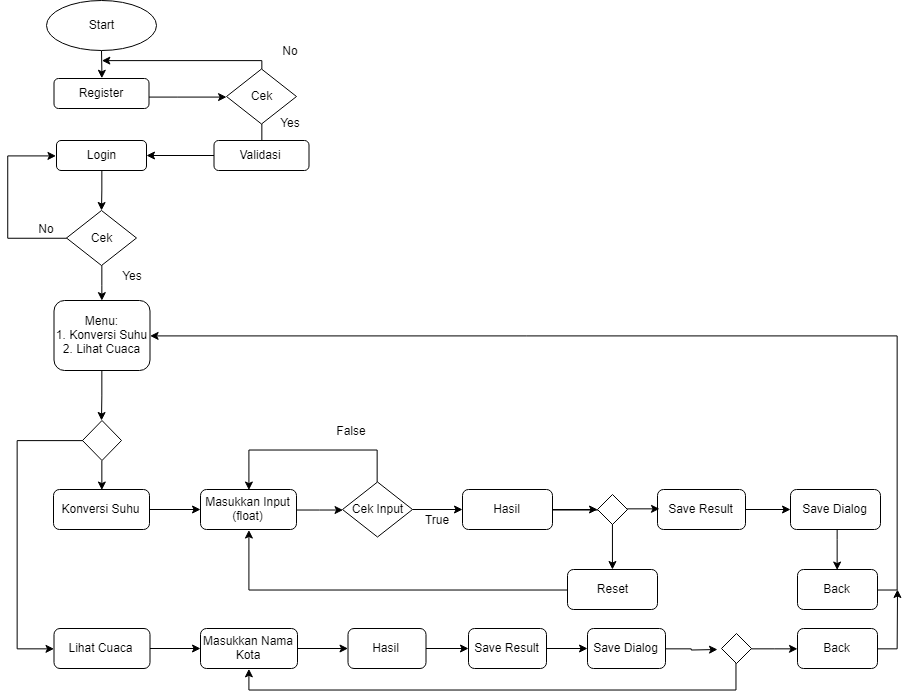
Usecase dan Activity diagram dibuat melalui draw.io, berikut usecase dan activity diagram nya :

**Usecase :**



Pengguna dapat Login terlebih dahulu, apabila belum mempunyai akun, maka bisa Register terlebih dahulu. Lalu pengguna dapat menggunakan fitur konversi suhu dan melihat kondisi cuaca. Pengguna juga dapat menyimpan kondisi cuaca yang dilihat.

**Activity Diagram**

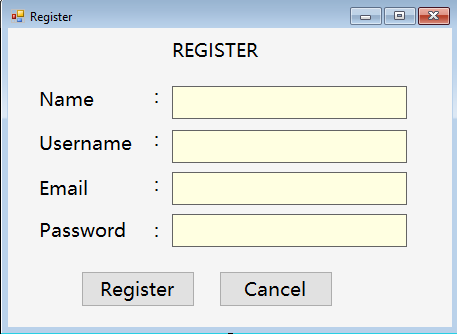


Pengguna baru harus register dan davalidasi terlebih dahulu, baru bisa login menggunakan username nya. Setelah itu, pengguna bisa login, tetapi sebelum bisa menggunakan fitur yang ada, dicek apakah username dan passwordnya sudah benar. Apabila sudah benar, pengguna baru bisa menggunakan fiturnya. Untuk menggunakan konversi suhu, pengguna memasukkan satu input float (Celcius/Fahrenheit/Kelvin/Reamur) saja, apabila pengguna tidak memasukkan input atau memasukkan lebih dari satu input, maka ada message box (“Masukkan satu input saja”), sesudah itu muncul hasilnya, pengguna bisa menyimpan hasil tersebut atau bisa reset input, dan kembali ke menu utama. Untuk melihat cuaca, pengguna perlu memasukkan nama kota, akan tampil hasilnya dan pengguna bisa menyimpan hasil kondisi tersebut, atau bisa kembali melihat kondisi cuaca kota lain. Apabila pengguna ingin kembali ke menu utama, pengguna bisa menggunakan tombol back.

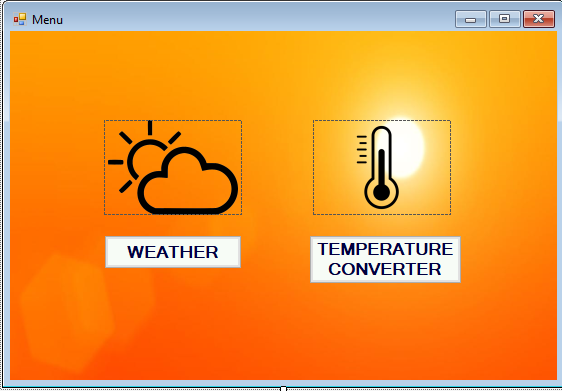
* 1. **Form Desain**
     1. **Login**



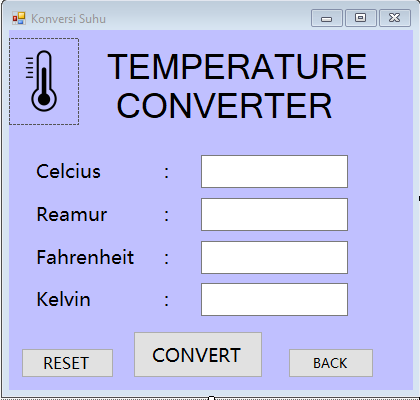
* + 1. **Register**



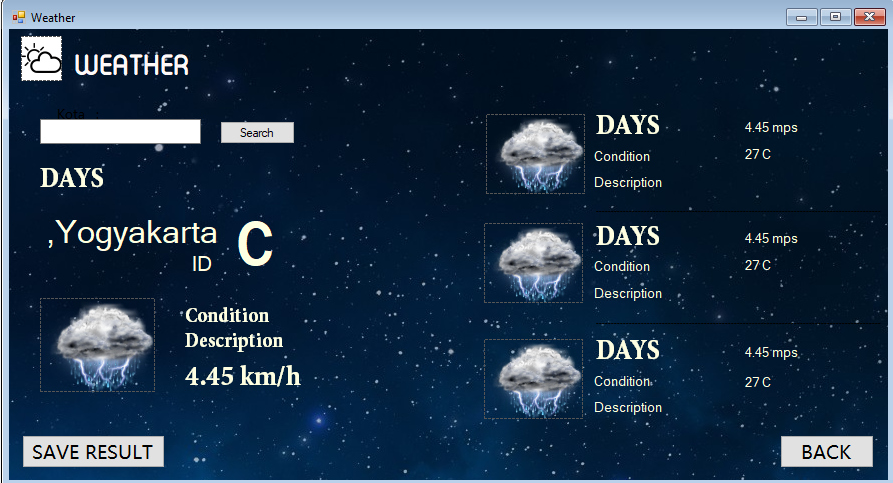
* + 1. **Menu**



* + 1. **Konversi Suhu**



* + 1. **Kondisi Cuaca**

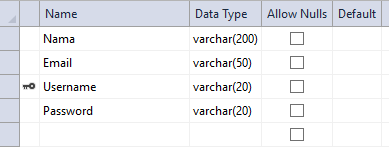


* 1. **Database**

Database dibuat dengan nama DatabaseUser.mdf diletakkan dalam local folder (F:\MATERI KULIAH\PBO\PRAKTIKUM PBO\PBO FINAL PROJECT\PBO FINAL PROJECT\DATABASEUSER.MDF), dengan Connection String (Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename="F:\MATERI KULIAH\ PBO\Praktikum PBO\PBO FINAL PROJECT\PBO FINAL PROJECT\ DatabaseUser.mdf";Integrated Security=True). Connection String ini nantinya berguna untuk mengkoneksikan update database dan command-commandnya seperti search untuk pencarian. Berikut T-SQL untuk pembuatan table dalam database

|  |
| --- |
| CREATE TABLE [dbo].[User] (  [Nama] VARCHAR (200) NOT NULL,  [Email] VARCHAR (50) NOT NULL,  [Username] VARCHAR (20) NOT NULL,  [Password] VARCHAR (20) NOT NULL,  CONSTRAINT [PK\_User] PRIMARY KEY ([Username]) ); |

Design Table Definition :



Username bertindak sebagai Primary key, karena tidak boleh ada yang username nya sama. Semua label diset not null.

* 1. **Penerapan Code**
  2. **Berpindah dari satu form ke form yang lain**

Setelah suatu tombol next (misal tombol Login, Register, atau Back) ditekan, maka form terkait akan secara otomatis ditutup dan form selanjutnya akan dibuka. Contohnya, jika pengguna menekan tombol Sign Up dari form login, maka secara otomatis form login akan tertutup dan form register akan terbuka. Berikut contoh source klik tombol register dari form login:

|  |
| --- |
| Close();  th = new Thread(openNewForm);  th.SetApartmentState(ApartmentState.STA);  th.Start(); |

Thread th adalah thread untuk membuka form baru. Close() untuk menutup form Login, dan th.Start() untuk memulai thread lalu membuka form register. Berikut method dari openNewForm:

|  |
| --- |
| private void openNewForm()  {  Application.Run(new Login());  } |

* 1. **Register**

Untuk method mengecek dan mengupdate data ke dalam database :

|  |
| --- |
| SqlConnection sqlcon = new SqlConnection(@"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=F:\MATERI KULIAH\PBO\Praktikum PBO\PBO FINAL PROJECT\PBO FINAL PROJECT\DatabaseUser.mdf;Integrated Security=True");  public void tambahData()  {  string q = "SELECT \* FROM [User] WHERE Username = '" + tbUsername.Text.Trim() + "';";  SqlDataAdapter sda2 = new SqlDataAdapter(q, sqlcon);  DataTable dataTable2 = new DataTable();  sda2.Fill(dataTable2);  if (dataTable2.Rows.Count >= 1)  {  MessageBox.Show("Registrasi Gagal. Username sudah terpakai. \n\tPakai Username yang lain");  }  else  {  sqlcon.Open();  SqlCommand cmd = sqlcon.CreateCommand();  cmd.CommandType = CommandType.Text;  cmd.CommandText = "INSERT INTO [USER] ([Nama], [Email], [Username], [Password]) VALUES ('" + tbNama.Text + "','" + tbEmail.Text + "','" + tbUsername.Text + "','" + tbPassword.Text + "')";  cmd.ExecuteNonQuery();  sqlcon.Close();  MessageBox.Show("Register berhasil");  Close();  th = new Thread(openNewForm);  th.SetApartmentState(ApartmentState.STA);  th.Start();  }  } |

Menggunakan SqlConnection untuk menghubungkan dengan database sql, dengan ConnectionString. q adalah query dan sqlcon adalah sqlconnection dengan connection data source. SqlDataAdapter sda sebagai adapter/penghubung, dengan parameter q dan sqlcon. Jika user menginput username yang sudah ada di database, if (dataTable2.Rows.Count >= 1) maka akan muncul message box dan akan kembali ke form register. Jika tidak, maka SqlConnection sqlcon terbuka dan menambah data dengan LINQ "INSERT INTO [USER] ([Nama], [Email], [Username], [Password]) VALUES ('" + tbNama.Text + "','" + tbEmail.Text + "','" + tbUsername.Text + "','" + tbPassword.Text + "')" , lalu sesudah ditambahkan, maka SqlConnection diputus dengan sqlcon.Close();. Jika berhasil register, maka muncul MessageBox dan pengguna form register tertutup dan akan dibawa ke form login.

Berikut event dari button Click Register bila sudah mengisi form:

|  |
| --- |
| private void btnAksi\_Click(object sender, EventArgs e)  {  if (tbNama.Text != "" && tbUsername.Text != "" && tbPassword.Text != && tbEmail.Text != "")  {  tambahData();  }  else  {  MessageBox.Show("Nama, Username, Email, dan Password harus diisi");  }  } |

Jika semua textbox sudah diisi, maka akan menuju method tambahData() untuk mengecek data dan mengupdate database, jika ada salah satu yang kosong, maka akan muncul MessageBox bahwa semua harus diisi.

* 1. **Login**

Untuk event klik Login, maka program akan mengecek apakah username dan password yang diisi benar atau salah.

|  |
| --- |
| private void btn\_Login\_Click(object sender, EventArgs e)  {  string query = "SELECT \* FROM [User] WHERE Username = '" + tb\_Username.Text.Trim() + "' and Password = '" + tb\_Password.Text.Trim() + "';";  SqlDataAdapter sda = new SqlDataAdapter(query, sqlcon);  DataTable dataTable = new DataTable();  sda.Fill(dataTable);  if (dataTable.Rows.Count == 1)  {  Akun akun = GetAkun(tb\_Username.Text);  Close();  th = new Thread(openNewForm);  th.SetApartmentState(ApartmentState.STA);  th.Start();  }  else  MessageBox.Show("Username atau Password Salah");    } |

Pengecekan dilakukan dengan SqlConnection dan SqlDataAdapter sda. sda diisi dengan dataTable inputan, dan mengecek apakah didalam table tersebut berisi username dan password yang tertera if (dataTable.Rows.Count == 1) . Jika terdapat dalam data table, maka akan menginstance Akun dengan method getAkun(), dan lanjut ke form menu. Berikut code getAkun()

|  |
| --- |
| public Akun GetAkun(string username)  {  Akun akun = new Akun();  string query = "SELECT \* FROM [User] WHERE Username = '" + tb\_Username.Text.Trim() + "' and Password = '" + tb\_Password.Text.Trim() + "';";  SqlCommand cmd = new SqlCommand(query, sqlcon);  cmd.Parameters.AddWithValue("@Username", tb\_Username.Text);  sqlcon.Open();  using (SqlDataReader reader = cmd.ExecuteReader())  {  while (reader.Read())  {  akun.nama = reader["Nama"].ToString();  akun.username = reader["Username"].ToString();  akun.email = reader["Username"].ToString();  }  sqlcon.Close();  }  return akun;  } |

Dengan menggunakan SqlData Reader, maka akan dapat membaca label pada table, lalu mensetnya sebagai akun.nama, akun.username, dan akun.email.

Untuk Constructor dari Akun sendiri ada empat, yaitu string nama, username, email, dan password.

|  |
| --- |
| public class Akun  {  public string nama { get; set; }  public string username { get; set; }  public string email { get; set; }  } |

Disinilah aplikasi ini menggunakan konsep OOP, dari Enkapsulasi dan didukung method getAkun() , agar mudah dikembangkan lebih lanjut. Pemanggilan method dapat dilakukan dengan Akun x = getAkun(). Akun tersebut akan dibaca oleh aplikas dengan info-info yang ada. Akun tersebut juga dapat diturunkan menjadi akun berbayar, bilamana ingin ditambahkan fitur-fitur yang baru yang hanya dapat diakses oleh akun tertentu.

* 1. **Konversi Suhu**

Di dalam menu terdapat 2 fitur, yaitu konversi suhu dan melihat cuaca, Untuk code nya saya seperti biasa hanya berpindah form dari form menu ke form konversi suhu atau dari form menu ke form cuaca. Pengguna juga bisa kembali ke menu utama dengan tombol back, dan logout dengan tombol log out.

Untuk program konversi suhu, dengan perbandingan Suhu yang sudah diketahui, yaitu C : R : F = 5 : 4 : 9 (+32), dan K = C+273;

Berikut algoritme dari Konversi Suhu:

1. Jika inputannya adalah C, maka hitung R, F, dan K
2. Jika inputannya adalah F, maka hitung C,R, dan K
3. Jika inputannya adalah R, maka hitung C, F, dan K
4. Jika inputannya adalah K, maka hitung C, R, dan F
5. Update Textbox

|  |
| --- |
| public void convert(float C, float F, float R, float K)  {  if (C != 0 && F == 0 && R == 0 && K == 0)  {  R = 4 \* C / 5;  F = (9 \* C / 5) + 32;  K = C + 273;  }  else if (C == 0 && F != 0 && R == 0 && K == 0)  {  C = 5 \* (F - 32) / 9;  R = 4 \* C / 5;  K = C + 273;  }  else if (C == 0 && F == 0 && R != 0 && K == 0)  {  C = 5 \* R / 4;  F = (9 \* C / 5) + 32;  K = C + 273;  }  else if (C == 0 && F == 0 && R == 0 && K != 0)  {  C = K - 273;  R = 4 \* C / 5;  F = (9 \* C / 5) + 32;  }  tb\_Celcius.Text = C.ToString();  tb\_Fahrenheit.Text = F.ToString();  tb\_Kelvin.Text = K.ToString();  tb\_Reamur.Text = R.ToString();  } |

Algoritme Keypress untuk setiap textbox (input):

1. Jika Input adalah digit dan karakter ‘.’ (tanda koma) , dan karakter ‘-‘ (tanda minus, maka event Handle = True
2. Jika Index Input karakter ‘.’ (tanda koma) > -1, maka event Handle = True
3. Jika Selection Start dari karakter ‘-‘ (tanda minus) tidak sama dengan 0, maka event Handle = True

|  |
| --- |
| private void tb\_Celcius\_KeyPress(object sender, KeyPressEventArgs e)  {  if (!char.IsControl(e.KeyChar) && !char.IsDigit(e.KeyChar) && (e.KeyChar != '.') && (e.KeyChar != '-'))  {  e.Handled = true;  }  if ((e.KeyChar == '.') && (sender as TextBox).Text.IndexOf('.')>-1)  {  e.Handled = true;  }  if ((e.KeyChar == '-') && (sender as TextBox).SelectionStart != 0)  {  e.Handled = true;  }  } |

Event Handle yang sama untuk tb\_Reamur\_KeyPress(), tb\_Fahrenheit\_KeyPress(), dan tb\_Kelvin\_KeyPress()

Untuk event button Convert Click, jika inputan sudah benar (hanya satu), maka lakukan convert, jika inputan lebih dari satu, maka muncul MessageBox.

|  |
| --- |
| private void button1\_Click(object sender, EventArgs e)  {  if (tb\_Celcius.Text == "")  C = 0;  else  C = float.Parse(tb\_Celcius.Text);  if (tb\_Fahrenheit.Text == "")  F = 0;  else  F = float.Parse(tb\_Fahrenheit.Text);  if (tb\_Reamur.Text == "")  R = 0;  else  R = float.Parse(tb\_Reamur.Text);  if (tb\_Kelvin.Text == "")  K = 0;  else  K = float.Parse(tb\_Kelvin.Text);  if ((tb\_Celcius.Text != "" && tb\_Reamur.Text == "" && tb\_Fahrenheit.Text == "" && tb\_Kelvin.Text == "") ||  (tb\_Celcius.Text == "" && tb\_Reamur.Text != "" && tb\_Fahrenheit.Text == "" && tb\_Kelvin.Text == "") ||  (tb\_Celcius.Text == "" && tb\_Reamur.Text == "" && tb\_Fahrenheit.Text != "" && tb\_Kelvin.Text == "") ||  (tb\_Celcius.Text == "" && tb\_Reamur.Text == "" && tb\_Fahrenheit.Text == "" && tb\_Kelvin.Text != ""))  {  convert(C, F, R, K);  }  else  MessageBox.Show("Masukkan Satu Input");  } |

Kondisi awal jika textbox tidak diisi, maka nilai suhu tersebut adalah 0. Dan Jika hanya salah satu textbox dari ke empat textbox inputan tersebut diisi, maka lakukan convert suhu, jika tidak, maka muncul MessageBox(“Masukkan satu input”).

Pengguna dapat mereset hasil dengan tombol Reset

|  |
| --- |
| private void button3\_Click(object sender, EventArgs e)  {  tb\_Celcius.Text = "";  tb\_Fahrenheit.Text = "";  tb\_Reamur.Text = "";  tb\_Kelvin.Text = "";  } |

Dengan menghapus semua nilai textbox dan mengembalikannya ke 0.

* 1. **Terhubung dengan API openweathermap**

Aplikasi ini menggunakan pendekatan Rest API di tingkat webclient. Disini saya menggunakan dua link request API, yaitu: <http://api.openweathermap.org/data/2.5/weather>?q= untuk weather dan <http://api.openweathermap.org/data/2.5/forecast>?q= untuk forecast. Contoh pemanggilan API yang berhasil:

Link Pemanggilan Weather: <http://api.openweathermap.org/data/2.5/weather?q=Yogyakarta&appid=542ffd081e67f4512b705f89d2a611b2&units=metric&cnt=6&mode=xml>

|  |
| --- |
| <current>  <city id="1621177" name="Yogyakarta">  <coord lon="110.36" lat="-7.8"/>  <country>ID</country>  <sun rise="2019-05-24T22:43:57" set="2019-05-25T10:27:13"/>  </city>  <temperature value="27.29" min="27.29" max="27.29" unit="celsius"/>  <humidity value="78" unit="%"/>  <pressure value="1012.07" unit="hPa"/>  <wind>  <speed value="7.72" name="Moderate breeze"/>  <gusts/>  <direction value="113.293" code="ESE" name="East-southeast"/>  </wind>  <clouds value="47" name="scattered clouds"/>  <visibility/>  <precipitation mode="no"/>  <weather number="802" value="scattered clouds" icon="03n"/>  <lastupdate value="2019-05-24T19:13:56"/>  </current> |

Link Pemanggilan Forecast:

<http://api.openweathermap.org/data/2.5/forecast?q=Yogyakarta&appid=542ffd081e67f4512b705f89d2a611b2&units=metric&cnt=6&mode=xml>

|  |
| --- |
| <weatherdata>  <location>  <name>Yogyakarta</name>  <type/>  <country>ID</country>  <timezone/>  <location altitude="0" latitude="-7.8012" longitude="110.3649" geobase="geonames" geobaseid="1621177"/>  </location>  <credit/>  <meta>  <lastupdate/>  <calctime>0.0052</calctime>  <nextupdate/>  </meta>  <sun rise="2019-05-24T22:43:56" set="2019-05-25T10:27:12"/>  <forecast>  <time from="2019-05-24T18:00:00" to="2019-05-24T21:00:00">...</time>  <time from="2019-05-24T21:00:00" to="2019-05-25T00:00:00">...</time>  <time from="2019-05-25T00:00:00" to="2019-05-25T03:00:00">...</time>  <time from="2019-05-25T03:00:00" to="2019-05-25T06:00:00">...</time>  <time from="2019-05-25T06:00:00" to="2019-05-25T09:00:00">...</time>  <time from="2019-05-25T09:00:00" to="2019-05-25T12:00:00">  <symbol number="802" name="scattered clouds" var="03n"/>  <precipitation/>  <windDirection deg="122.922" code="ESE" name="East-southeast"/>  <windSpeed mps="7.65" name="Moderate breeze"/>  <temperature unit="celsius" value="27.56" min="27.56" max="27.56"/>  <pressure unit="hPa" value="1010.79"/>  <humidity value="78" unit="%"/>  <clouds value="scattered clouds" all="28" unit="%"/>  </time>  </forecast>  </weatherdata> |

Sesuai dengan variable xml yang ada, langkah selanjutnya adalah Mengkonstruksikan variable-variabel Weather kedalam WeatherInfo.cs dan Forecast kedalam WeatherForecast.cs

|  |
| --- |
| class WeatherInfo  {  public class weather  {  public int id { get; set; }  public string main { get; set; }  public string description { get; set; }  public string icon { get; set; }  }  public class main  {  public double temp { get; set; }  }  public class wind  {  public double speed { get; set; }  }  public class sys  {  public string country { get; set; }  }  public class root  {  public string name { get; set; }  public sys sys { get; set; }  public double dt { get; set; }  public wind wind { get; set; }  public main main { get; set; }  public List<weather> weather { get; set; }  }  } |

|  |
| --- |
| class weatherForecast  {  public city city { get; set; }  public List<list> list { get; set; }  }  public class weather  {  public string main { get; set; }  public string description { get; set; }  public string icon { get; set; }  }  public class temp  {  public double day { get; set; }  }  public class city  {  public string name { get; set; }  }  public class list  {  public double dt { get; set; }  public double speed { get; set; }  public temp temp { get; set; }  public List<weather> weather { get; set; }  } |

Disini hanya perlu mengkonstruksikan objek-objek yang ingin digunakan beserta tipenya. Lalu Mempresentasikannya kedalam Form dengan membuat method getWeather() dan getForecast()

|  |
| --- |
| void getWeather(string city)  {  using (WebClient web = new WebClient())  {  string url = string.Format("http://api.openweathermap.org/data/2.5/weather?q={0}&appid={1}&units=metric&cnt=6", city, APPID);  var json = web.DownloadString(url);  var result = JsonConvert.DeserializeObject<WeatherInfo.root>(json);  WeatherInfo.root output = result;  lbcityName.Text = string.Format("{0}", output.name);  lbcountryName.Text = string.Format("{0}", output.sys.country);  lbSuhu.Text = string.Format("{0}\u00B0"+"C",output.main.temp);  lbDescription.Text = string.Format("{0}", output.weather[0].description);  lbCondition.Text = string.Format("{0}", output.weather[0].main);  lbWind.Text = string.Format("{0} km/h", output.wind.speed);  pictureBox2.Image = setIcon(output.weather[0].icon);  }  } |

Method getWeather(string city) berguna untuk menampilkan kondisi cuaca. Contohnya getWeather(“Surakarta”) akan menampilkan kondisi cuaca di kota Surakarta. Dengan menggunakan WebClient dan url yang ada, maka label-label di Form dapat menyesuaikan label-label di url request. Kondisi cuaca yang ditampilkan menyesuaikan label-label yang ada di API openweathermap. Untuk \u00B0 berarti unicode lambang derajat. Disini juga menampilkan picture box yang bisa melambangkan kondisi cuaca, contohnya hujan ringan, atau terang, atau berkabut. Method getWeather() ini akan menampilkan suhu, kecepatan angina, kondisi dan deskripsinya di suatu kota pada hari ini.

|  |
| --- |
| void getForecast(string city)  {  int day = 5;  string url = string.Format("http://api.openweathermap.org/data/2.5/forecast/daily?q={0}&units=metric&cnt={1}&APPID={2}", city, day, APPID);  using (WebClient web = new WebClient())  {  var json = web.DownloadString(url);  var Object = JsonConvert.DeserializeObject<weatherForecast>(json);  weatherForecast forecast = Object;  lbDay.Text = string.Format("{0}", GetDate(forecast.list[0].dt).DayOfWeek);  lbDay1.Text = string.Format("{0}", GetDate(forecast.list[1].dt).DayOfWeek);  lbCondiiton1.Text = string.Format("{0}", forecast.list[1].weather[0].main);  lbDescription1.Text = string.Format("{0}", forecast.list[1].weather[0].description);  lbSuhuDay1.Text = string.Format("{0} \u00B0" + "C", forecast.list[1].temp.day);  lbWind1.Text = string.Format("{0} km/h", forecast.list[1].speed);  pb2.Image = setIcon(forecast.list[1].weather[0].icon);  lbDay2.Text = string.Format("{0}", GetDate(forecast.list[2].dt).DayOfWeek);  lbCondition1.Text = string.Format("{0}", forecast.list[2].weather[0].main);  lbDescription2.Text = string.Format("{0}", forecast.list[2].weather[0].description);  lbSuhuDay2.Text = string.Format("{0} \u00B0" + "C", forecast.list[2].temp.day);  lbWind2.Text = string.Format("{0} km/h", forecast.list[2].speed);  pb3.Image = setIcon(forecast.list[2].weather[0].icon);  lbDay3.Text = string.Format("{0}", GetDate(forecast.list[3].dt).DayOfWeek);  lbCondition3.Text = string.Format("{0}", forecast.list[3].weather[0].main);  lbDescription3.Text = string.Format("{0}", forecast.list[3].weather[0].description);  lbSuhuDay3.Text = string.Format("{0} \u00B0" + "C", forecast.list[3].temp.day);  lbWind3.Text = string.Format("{0} km/h", forecast.list[3].speed);  pb4.Image = setIcon(forecast.list[3].weather[0].icon);  }  } |

Method getForecast(string city) berguna untuk menampilkan ramalan cuaca 3 hari kedepan. Contohnya getForecast(“Surakarta”) akan menampilkan ramalan cuaca di kota Surakarta 3 hari kedepan. Dengan menggunakan WebClient dan url yang ada, maka label-label di Form dapat menyesuaikan label-label di url request. Kondisi cuaca yang ditampilkan menyesuaikan label-label yang ada di API openweathermap. Untuk \u00B0 berarti unicode lambang derajat. Disini juga menampilkan icon (picture box) yang bisa melambangkan kondisi cuaca, contohnya hujan ringan, atau terang, atau berkabut. Method getForecast() ini akan menampilkan suhu, kecepatan angina, kondisi dan deskripsi di suatu kota pada tiga hari kedepan.

* 1. **Menyimpan Hasil**

Untuk menyimpan hasil konversi suhu:

|  |
| --- |
| private void button4\_Click(object sender, EventArgs e)  {    SaveFileDialog saveFile = new SaveFileDialog();  saveFile.Title = "Save File";  saveFile.DefaultExt = "txt";  saveFile.InitialDirectory = @"Desktop:\";  saveFile.Filter = "Text Files (\*.txt)|\*.txt|All files(\*.\*)|\*.\*";  saveFile.RestoreDirectory = true;  saveFile.FilterIndex = 2;  if (saveFile.ShowDialog() == DialogResult.OK)  {  var path = saveFile.FileName;  StreamWriter str = new StreamWriter(path);  str.WriteLine("Celcius : {0} \u00B0C", tb\_Celcius.Text);  str.WriteLine("Reamur : {0} \u00B0R", tb\_Reamur.Text);  str.WriteLine("Fahrenheit : {0} \u00B0F", tb\_Fahrenheit.Text);  str.WriteLine("Kelvin : {0} K", tb\_Kelvin.Text);  str.Close();  }  } |

Dengan menggunakan library System.IO, saveFileDialog bertujuan untuk menampilkan dialog penyimpanan file ke suatu folder yang diinginkan tempatnya. Judul dari dialognya adalah “Save File”, default extension nya \*.txt, initial directory saya buat di desktop. Restore directory untuk menyimpan directory akhir sebagai initial directory. StreamWriter disini berguna untuk menulis suatu stream kedalam file tersebut. Jadi, setelah menngklik save button, maka muncul save dialog dan directorynya, program akan membuat file baru (path), disimpan kedalam folder yang diinginkan, setelah tombol ok/save, maka StreamWriter akan menuliskan hasil konversi suhu tersebut. Jika sudah selesai maka streamwriter akan di close str.Close();

Begitu pula dengan code untuk menyimpan hasil cuaca dan ramalan cuaca:

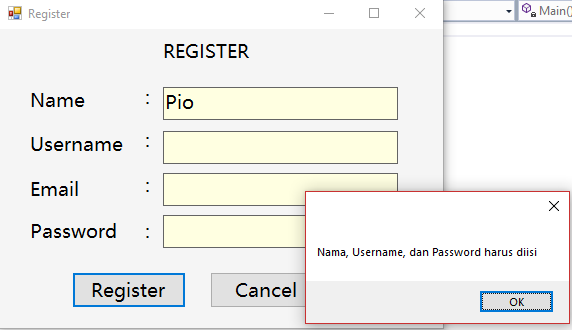
|  |
| --- |
| private void btnSave\_Click(object sender, EventArgs e)  {  SaveFileDialog saveFile = new SaveFileDialog();  saveFile.Title = "Save File";  saveFile.DefaultExt = "txt";  saveFile.InitialDirectory = @"Desktop:\";  saveFile.Filter = "Text Files (\*.txt)|\*.txt|All files(\*.\*)|\*.\*";  saveFile.RestoreDirectory = true;  saveFile.FilterIndex = 2;  if (saveFile.ShowDialog() == DialogResult.OK)  {  var path = saveFile.FileName;  StreamWriter str = new StreamWriter(path);  str.WriteLine("City : {0}", lbcityName.Text);  str.WriteLine("Country : {0}", lbcountryName.Text);  str.WriteLine("Temperature : {0} \u00B0C", lbSuhu.Text);  str.WriteLine("Condition : {0}", lbCondition.Text);  str.WriteLine("Description : {0}", lbDescription.Text);  str.Close();  }  } |

Proses yang terjadi saat klik tombol save: memunculkan save dialog dan directory nya, program akan membuat file baru (path), disimpan kedalam folder yang diinginkan, setelah tombol ok/save, maka StreamWriter akan menuliskan hasil cuaca dan ramalan ruaca tersebut. Jika sudah selesai maka streamwriter akan di close str.Close();

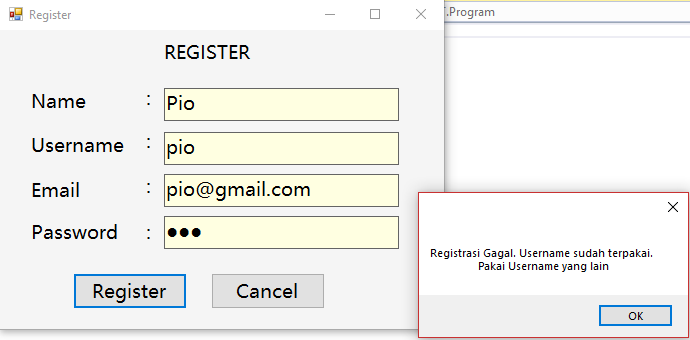
* 1. **Pengujian Aplikasi**

**Login dan Register**

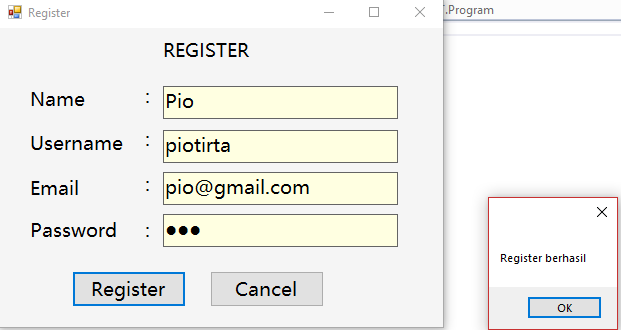
1. Register gagal:
   1. Karena pengisian tidak lengkap (ada yang NULL) (Register.cs line 56-66):



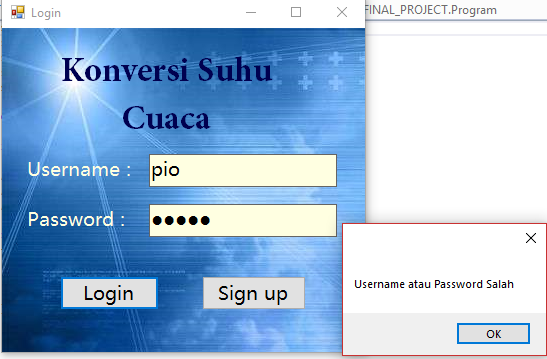
* 1. Karena username sudah terpakai (Register.cs line 35-38):



1. Register berhasil :

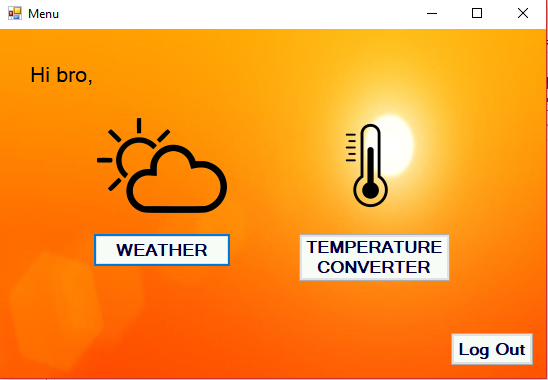


1. Login gagal



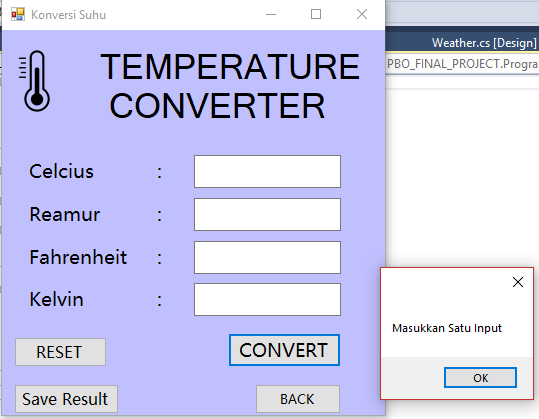
1. Login berhasil

Langsung diarahkan ke menu tanpa ada message box

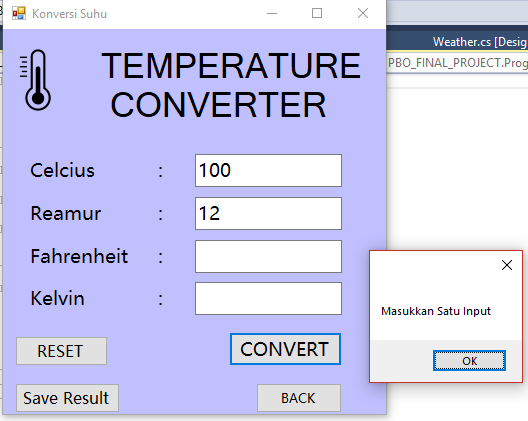


**Konversi Suhu**

1. Konversi Suhu gagal karena tidak ada input

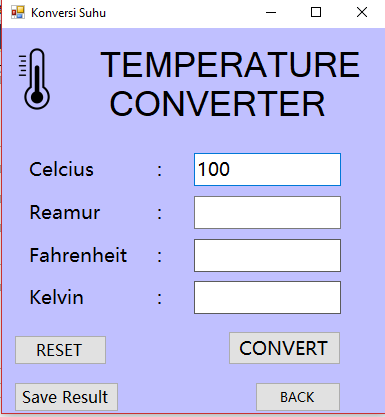


1. Konversi Suhu gagal karena ada lebih dari 1 input

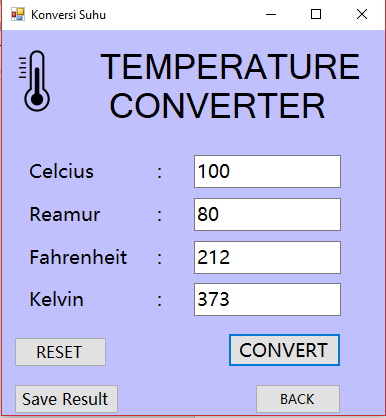


1. Konversi Suhu berhasil

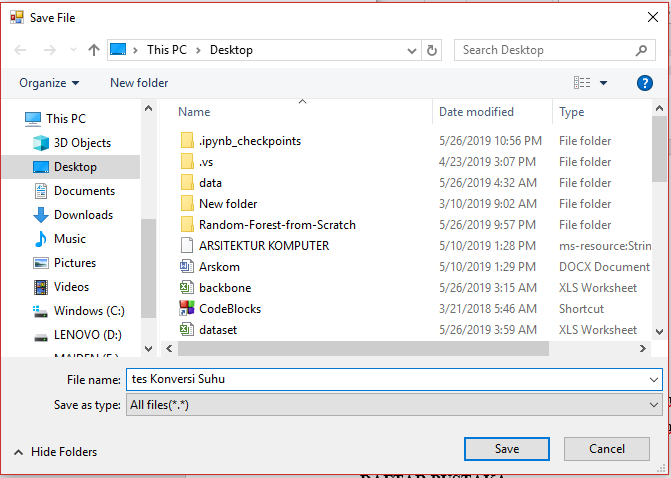
Input:



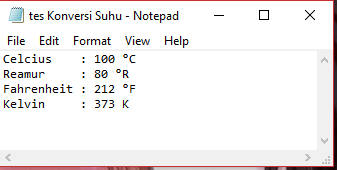
Hasil:



1. Menyimpan Hasil konversi suhu

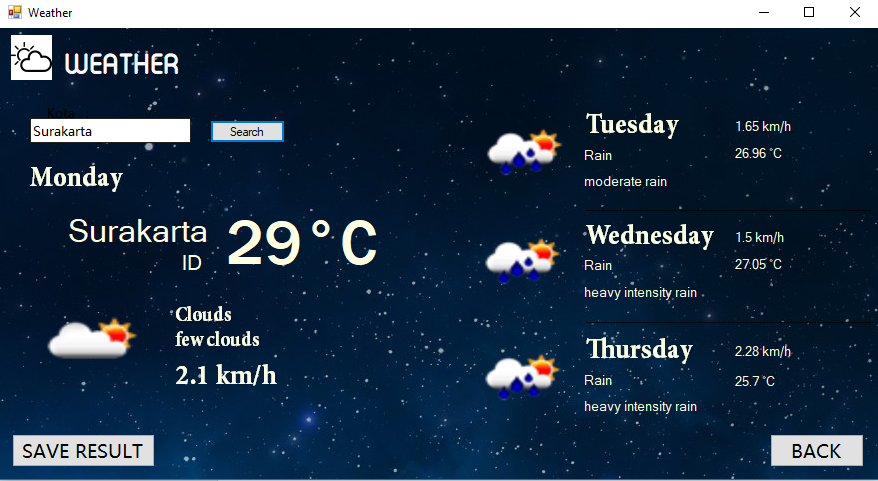


Hasil tes Konversi Suhu.txt:

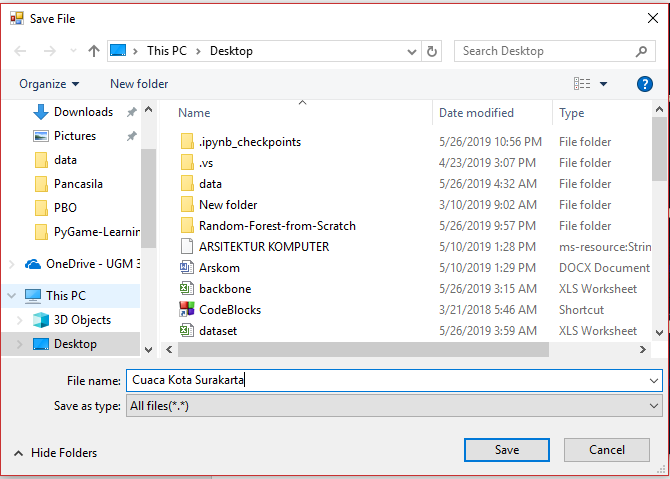


**Ramalan Cuaca**

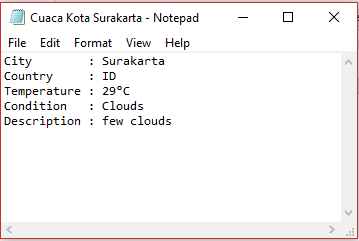
* + 1. Kondisi cuaca dan Ramalan cuaca kota Surakarta



* + 1. Menyimpan Hasil



Hasil Cuaca Kota Surakarta.txt:



1. **KESIMPULAN**

Aplikasi ini bertujuan untuk mengkonversi suhu dan melihat cuaca. Pengguna juga dapat menyimpan hasil tersebut kedalam file .\*txt. Aplikasi ini dibuat dengan Pilar OOP untuk mempermudah pengembangannya kedepan, didukung dengan database dan Web API openweathermap.

**DAFTAR PUSTAKA**

Modul Praktikum Pemrograman Berorientasi Objek

<https://openweathermap.org>

<https://docs.microsoft.com/en-us/nuget/quickstart/install-and-use-a-package-in-visual-studio>

<https://stackoverflow.com/questions/53288491/c-sharp-textbox-allow-minus-sign-only-in-the-beginning-of-string>

**LAMPIRAN**

Source Code: <https://github.com/profspio21/pbota>